National Defence / Défense nationale

MULTIPLE-SYMBOL DIFFERENTIAL DETECTION OF DPSK WITH FREQUENCY OFFSET (U)

by

Martin Turgeon
Communications and Navigation Section
Electronics Division

DEFENCE RESEARCH ESTABLISHMENT OTTAWA
TECHNICAL NOTE 92-5

PCN
D6470

December 1991
Ottawa

93-01768/

Abstract

A ground terminal (GT) simulator for an extremely high frequency (EHF) satellite communications (SATCOM) system is being developed as part of the EHF SATCOM project at the Defence Research Establishment Ottawa (DREO). The GT demodulator uses *multiple-symbol differential detection of DPSK* (MSDPSK), a demodulation technique which estimates the phase reference of a DPSK modulated, burst signal over a sequence of N symbols. The bit error rate (BER) performance is investigated and the simulation results are presented. The BER performance of MSDPSK is shown to approach that of coherently detected DPSK (DEPSK) for large N. The effects of frequency offset are also investigated. For a specific amount of frequency uncertainty, there is an optimum sequence size N that yields the best BER performance. For example, given a burst size of 13 bits and a frequency offset up to 1/50 of the bit rate, the optimum sequence size is in the range of 3 to 5 bits.

Résumé

Un simulateur d'une station terrestre pour un système de communication par satellite (SATCOM) à fréquence extrêmement haute (EHF) est présentement développé grâce au projet EHF SATCOM au Centre de recherches pour la défense, Ottawa. Le démodulateur de la station terrestre utilise la détection différentielle de DPSK sur multiples symboles (MSDPSK). Cette technique de démodulation estime la phase de référence de rafales de signaux modulés avec DPSK pour une séquence de N symboles. La performance en terme du taux d'erreur des bits est examinée et les résultats de simulation sont présentés. Il est montré que cette performance de MSDPSK approche celle de la détection cohérente de DPSK (DEPSK) pour une grande valeur de N. Les effets d'un décalage de fréquence sont aussi examinés. Pour une valeur spécifique de décalage de fréquence, il existe une séquence de longueur optimale N qui donne le meilleur taux d'erreur. Etant donnée une longueur de rafales de 13 bits et un décalage de fréquence allant jusqu'à 1/50 du taux de transmission, la longueur optimale de la séquence est de 3 à 5 bits.

Executive Summary

A ground terminal (GT) simulator for an extremely high frequency (EHF) satellite communications (SATCOM) system is being developed as part of the EHF SATCOM project at the Defence Research Establishment Ottawa (DREO). The GT demodulator uses *multiple-symbol differential detection of DPSK* (MSDPSK), a demodulation technique which estimates the phase reference of a DPSK modulated, burst signal over a sequence of N symbols. The technique improves upon the bit error rate (BER) performance of *differential phase-shift keying* (DPSK) and approaches that of *differentially encoded phase-shift keying* (DEPSK).

MSDPSK extends the usual two-symbol observation interval of DPSK demodulation to a sequence of N symbols so that the carrier phase reference may be approximated. With the phase reference assumed constant over N symbols, a decision on the $N - 1$ symbols can then be made. Since DPSK demodulation is based on a two-symbol estimate of the phase reference and DEPSK assumes an infinite number of symbols, the MSDPSK technique is expected to provide a BER between the two more conventional demodulation techniques.

The bit error rate (BER) performance is investigated and the simulation results are presented. The BER performance of MSDPSK is shown to outperform DPSK in terms of BER performance. In fact, the MSDPSK scheme is shown to approach that of coherently detected DPSK (DEPSK) with N as small as 5.

The effects of frequency offset are also investigated. The results are presented for a burst-mode transmission with a variable sequence or window size for the phase reference approximation. Simulation results indicate that a specific frequency offset is associated with an optimum sequence or window size in terms of BER performance. For a long sequence size, the BER performance degradation due to frequency offset increases very rapidly. For example, given a burst size of 13 bits and a frequency offset up to 1/50 of the bit rate, the optimum sequence size is in the range of 3 to 5 bits.

Table of Contents

Abstract iii

Résumé iii

Executive Summary v

Table of Contents vii

List of Figures xi

1 Introduction 1
- 1.1 The Aim 1
- 1.2 Phase-Shift Keying (PSK) Schemes with Coherence Nuances 1
- 1.3 System Constraints 2
- 1.4 Phase Reference Estimation Over a 13-Bit Burst 4

2 Signal Characteristics 4
- 2.1 PSK and DPSK Modulation 4
- 2.2 Decision Rule Statistic 5
- 2.3 Differential Decoding 7
- 2.4 Approaches to Variable Sequence Size 7

3 Bit Error Probability Performance 8
- 3.1 Theoretical BER Performance 8
- 3.2 Simulation Results without Frequency Offset 10
- 3.3 Simulation Results with Frequency Offset 11
- 3.4 Future Work 15

4 Conclusions 15

Appendix A: Derivation of Equation (9) 16

Appendix B: SPW System Implementation 18
- B.1 General 18
- B.2 Block Implementation 18
- B.3 Implementation Details 18

Appendix C: Batch Execution 32
- C.1 General 32
- C.2 Batch Simulations at the Design Level 32

C.3 Batch Simulations for a Completed System 33
C.4 Example of a Batch File Macro at the Design Level 37

References 40

List of Figures

Fig. 1. Bit error probability versus E_b/N_o for PSK, DEPSK, and DPSK 3
Fig. 2a. DPSK modulator block diagram 6
Fig. 2b. Optimum DPSK receiver block diagram 6
Fig. 3a. BER vs E_b/N_o for the upper bounds of MSDPSK as a function of the window size 9
Fig. 3b. BER vs E_b/N_o for $N = 3$ as a function of the burst size (no frequency offset) . 9
Fig. 3c. BER vs E_b/N_o as a function of the burst size (no frequency offset) 12
Fig. 3d. BER vs E_b/N_o as a function of the window size (no frequency offset) 12
Fig. 4a. BER vs E_b/N_o with $b = 13$ and $N = 5$ as a function of δ in degree per bit . 13
Fig. 4b. BER vs E_b/N_o with $b = N = 13$ as a function of δ in degree per bit 13
Fig. 5a. BER vs E_b/N_o with $\delta = 4.32$ degree/bit and $b = 13$ as a function of the window size 14
Fig. 5b. BER vs E_b/N_o with $\delta = 5.76$ degree/bit and $b = 13$ as a function of the window size 14
Fig. 5c. BER vs E_b/N_o with $\delta = 7.20$ degree/bit and $b = 13$ as a function of the window size 14
Fig. 6. MSDPSK low-pass equivalent system 19
Fig. 7. Differential encoder 20
Fig. 8. Phase modulator 21
Fig. 9. Channel with Rayleigh fading (optional) and AWGN 22
Fig. 10. Random phase channel effect for each burst 23
Fig. 11. Phase reference estimator for each window 24
Fig. 12. Symbol phase and phase reference comparator 25
Fig. 13a. Bit reference adjustment for decoded bit stream 26
Fig. 13b. Bit reference adjustment and synchronisation of data bit stream 26
Fig. 14. Bit error rate calculator 27
Fig. 15. Detail block of hold signal 29
Fig. 16. Detail block of delay 29
Fig. 17a. Timing control specified by the start of each burst 30
Fig. 17b. Timing control specified by the start of each window 30
Fig. 17c. Timing control specified by the last bit of each window 30

1 Introduction

1.1 The Aim

A ground terminal (GT) simulator for an extremely high frequency (EHF) satellite communications (SATCOM) system is being developed as part of the EHF SATCOM project at the Defence Research Establishment Ottawa (DREO). This report focuses on the ground terminal demodulator. More specifically, the report focuses on a differentially-coherent detection technique which improves upon the bit error rate (BER) performance of *differential phase-shift keying* (DPSK) and approaches that of *differentially encoded phase-shift keying* (DEPSK). The technique is referred to as *multiple-symbol differential detection of DPSK* (MSDPSK). The aim of the report is twofold. First, it is to investigate the applicability of this technique for the ground terminal demodulator. Second, it is to analyze the effects of frequency uncertainty coupled with the number of symbols used to approximate phase coherence. Simulation results are presented and compared to well-known bounds.

1.2 Phase-Shift Keying (PSK) Schemes with Coherence Nuances

Phase-shift keying (PSK) is a widely used modulation scheme since bandwidth efficiency, constant envelope, and good noise immunity make this scheme, or one of its derivatives, particularly attractive for SATCOM. Binary PSK uses two antipodal signals and hereafter, will be the assumed signal set. During the transmission, a random phase is added to the signal due to the unknown delay of the channel. At the receiver, the baseband information is recovered by comparing each received symbol with an estimated phase reference. Phase reference estimation is the most complex demodulation task for a PSK scheme. This section presents a brief overview of PSK schemes in terms of the phase reference estimation. In addition, some clarification is provided on the nuances associated with coherence and differential encoding/detection with the following commonly used techniques:

a. PSK modulation and demodulation;

b. DPSK modulation;

c. DEPSK demodulation; and

d. DPSK demodulation.

For a PSK demodulator, a reference signal with the same phase as the carrier of the received signal is required. The reference and the received signals are mixed to recover the baseband signal. When an in-phase reference signal is used (i.e. a perfect reference signal), the term *coherent detection* is applicable. A carrier-recovery circuit in the form of a phase-locked loop (PLL) is typically used to track the carrier phase; however, the PLL introduces a 180° phase

uncertainty for binary PSK. Clearly, this phase ambiguity or rotation may cause serious detection errors.

The DPSK modulation technique consists on having the information carried by the phase transition between two consecutive symbols rather than the phase itself, and thereby remove phase ambiguity at the receiver. To recover the information, one of two traditional demodulation schemes is typically used: coherent detection, DEPSK, and differentially-coherent detection, DPSK.

DEPSK demodulation, which assumes a perfect reference signal, is the optimum demodulation scheme for DPSK modulation. Each symbol phase is compared to the phase reference to determine the differentially encoded bit. Each differentially encoded bit is then compared to the previous one to determine the uncoded bit polarity. The dependence between consecutive bits generally results in paired errors. Hence, the price to be paid (with respect to PSK) is a doubling of the number of errors which translates into a performance degradation of 0.55 dB in signal-to-noise ratio (SNR) at a BER of 10^{-3}. Fig. 1 shows the relative BER performance of DEPSK with respect to PSK. Note that DEPSK is sometimes referred to as *ideally-coherent detection of DPSK*.

The DPSK demodulation scheme requires a reference signal that is in-frequency but not necessarily in-phase with the received carrier [6]. An assumption made is that the received carrier phase reference is constant over at least two consecutive symbols. After down-conversion, the decoded symbol polarity is recovered by comparing two consecutive baseband symbols. In other words, the previous symbol becomes the phase reference for the current symbol and in that sense, the detection may be referred to as *differentially-coherent* [8]. Because the previous symbol may be a noisy reference rather than a tracked, noise-free carrier-phase reference, a DPSK demodulator is inferior in terms of BER to the DEPSK scheme; for example, DPSK suffers a 0.63 dB degradation with respect to DEPSK at a BER of 10^{-3} as shown in Fig. 1. When a perfect reference signal (i.e. carrier phase tracking) is not practical, such as with frequency-hopped systems, DPSK becomes a viable demodulation scheme.

1.3 System Constraints

At the demodulator, the received signal consists of bursts of b symbols which are modulated using DPSK. Each burst consists of a phase reference symbol, followed by $b - 1$ data symbols. The burst-mode transmission prevents the use of traditional, coherent demodulation schemes. For the receiver to use a PLL, some acquisition time must be provided for the PLL to attain its steady-state performance. In a burst mode, the PLL acquisition time requirement cannot be met unless a preamble of significant length is used on each burst. The nature of burst transmission also prevents maintaining phase coherence from one burst to the next as the initial phase of each burst is effectively a random variable. In an attempt to bridge the gap between DPSK and DEPSK, an alternate demodulation scheme which performs phase estimation over multiple symbols was investigated.

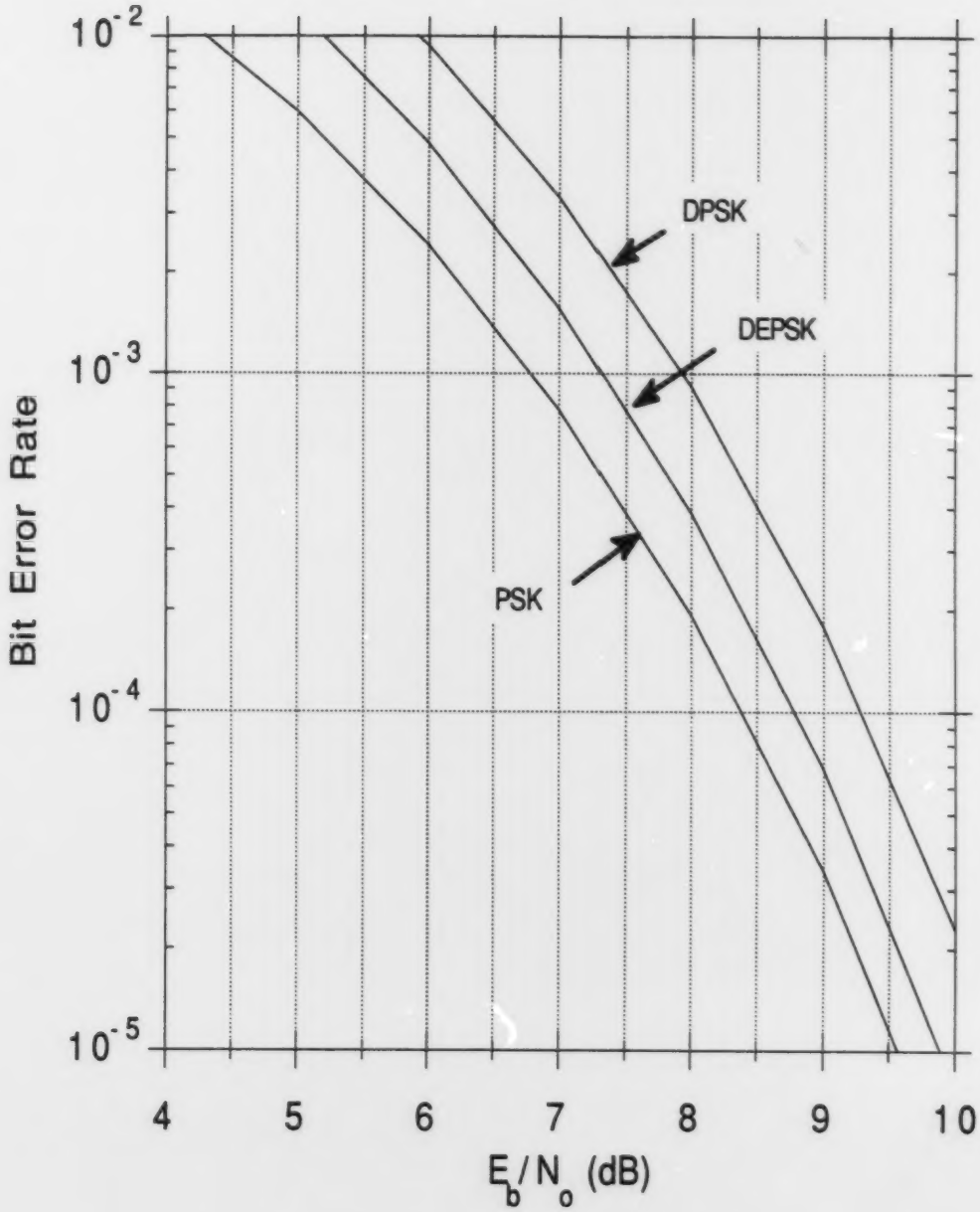


Fig. 1. Bit error probability versus E_b / N_o for PSK, DEPSK, and DPSK.

1.4 Phase Reference Estimation Over a 13-Bit Burst

The demodulation technique studied in this report extends the usual two-symbol observation interval of DPSK demodulation to a sequence of N symbols, where $N \leq b$, so that the carrier phase reference may be approximated [1]. With the phase reference assumed constant over N symbols, a decision on the $N - 1$ symbols can then be made. This demodulation technique is referred to as *multiple-symbol differential detection of DPSK* (MSDPSK) and is reviewed in Section 2. Since DPSK demodulation is based on a two-symbol estimate ($N = 2$) of the phase reference and DEPSK assumes an infinite number of symbols ($N = \infty$), the MSDPSK technique is expected to provide a BER between the two more conventional demodulation techniques.

This report extends the studies on the effects of phase error and frequency uncertainty on DPSK [2-5]. The effects of frequency offset on MSDPSK, obtained via simulation, are presented in Section 3. The effects of bit timing error on the BER performance will be presented in a later report.

2 Signal Characteristics

2.1 PSK and DPSK Modulation

Let the transmitted signal in the interval $kT_b \leq t \leq (k + 1)T_b$ be defined by

$$s(t) = \sqrt{2P}\, e^{j(2\pi f_c t + \phi_k)} \tag{1}$$

where P denotes the constant signal power, f_c is the carrier frequency, T_b is the bit duration, and ϕ_k denotes the transmitted symbol phase. For antipodal signals, ϕ_k takes on one of two equiprobable values, 0° or 180°. Without loss of generality, ϕ_k is defined for PSK modulation by

$$\phi_k \triangleq \begin{cases} 0^\circ, & \text{when } \alpha_k = 0 \\ 180^\circ, & \text{when } \alpha_k = 1 \end{cases} . \tag{2}$$

As discussed in Section 1, phase ambiguity may be resolved by differentially encoding the data bits, α's. The k^{th} binary encoded bit, β_k, is such that

$$\beta_k = \alpha_k \oplus \beta_{k-1} \tag{3}$$

where β_0 is set to 0 and $\oplus$ is the exclusive-or operation. For DPSK modulation, ϕ_k in (1) is associated with a differentially encoded bit, β_k, and as for PSK modulation, the relationship may be arbitrarily fixed. Without loss of generality, ϕ_k is defined for DPSK modulation by

$$\phi_k \triangleq \begin{cases} 0^\circ, & \text{when } \beta_k = 0 \\ 180^\circ, & \text{when } \beta_k = 1 \end{cases} . \tag{4}$$

In words, an uncoded bit of value 1 corresponds to a 180° phase transition between two consecutive transmitted symbols and a 0 results in no phase transition.

A simplified modulation block diagram is shown in Fig. 2a. It includes differential encoding and an up-conversion to the carrier frequency, f_c.

2.2 Decision Rule Statistic

Through the channel, the transmitted signal is corrupted with additive white Gaussian noise (AWGN). The received signal is described by [1]

$$r(t) = s(t)e^{j\theta} + n(t) \tag{5}$$

where θ, an arbitrary phase introduced by the channel, is assumed stationary and uniformly distributed in the interval $[\pm 180^\circ]$. After being mixed to baseband, the received signal is integrated and sampled at the end of each T_b interval, as shown in the optimum DPSK receiver block diagram in Fig. 2b [6]. The integrator output is reset to zero after sampling. The sampled signal, r_k, is described in the interval $kT_b \le t \le (k + 1)T_b$ by

$$r_k = s_k e^{j\theta} + n_k \tag{6}$$

where s_k is the k^{th} symbol and n_k is a sample of AWGN with variance

$$\sigma_n^2 = \frac{2N_o}{T_b} \tag{7}$$

where N_o is the single-sided noise spectral density.

Consider now a sequence of N received PSK symbols. For detection, the maximum-likelihood approach sequence estimation (MLSE) as given in [1] is followed. The MLSE for

[1] The channel attenuation is neglected as it does not affect the decision rule statistic of PSK or one of its derivatives.

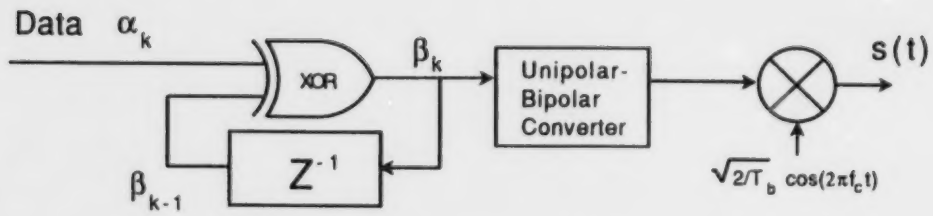


Fig. 2a. DPSK modulator block diagram.

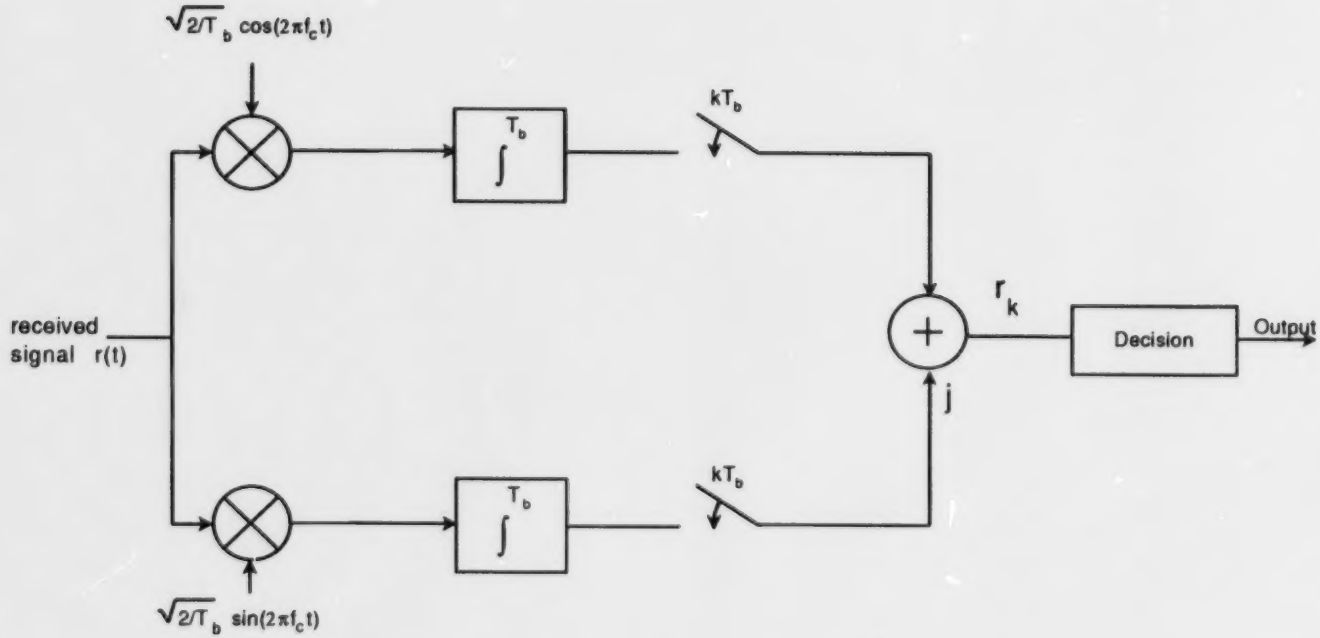


Fig. 2b. Optimum DPSK receiver block diagram.

PSK results in the decision rule statistic

choose the $\hat{\phi}_k$'s, the estimated phases, so that

$$\eta = \left| \sum_{i=0}^{N-1} r_{k-i} e^{-j\hat{\phi}_{k-i}} \right|^2 = |R|^2 \text{ is maximum} \tag{8}$$

where R is the sequence (resulting) vector. For a binary scheme, note that the exponential factors in (8) take the values ± 1. In words, the rule in (8) corresponds to choosing, for each signal vector, either the received vector itself or its 180°, rotated version. The phase of R is the approximated phase reference for each symbol in the received sequence. As discussed earlier, PSK is susceptible to phase ambiguity and so is the decision rule in (8). There are two sequence vectors, R and R', with the same maximum magnitude and a phase difference of 180°. Because of this uncertainty, DPSK is selected as the modulation scheme.

2.3 Differential Decoding

The decision rule in (8) still applies when differential encoding is applied at the transmitter; however, the phase estimates are now for differentially encoded phases. There are still two sequence vectors with the same maximum magnitude. One vector is associated with the original symbol in the sequence while the other vector is related to the 180° rotation of this symbol. Without loss of generality, the first sequence symbol will not be inverted. The statistic in (8) may then be rewritten as

$$\eta = \left| r_k e^{-j\hat{\phi}_k} + r_{k-1} e^{-j(\hat{\phi}_{k-1})} + r_{k-2} e^{-j(\hat{\phi}_{k-2})} + \ldots + r_{k-N+1} \right|^2 \tag{9}$$

The equivalence between the statistic in (9) and that of equ. (15) in [1] is given in Appendix A. The statistic in (9) relies on estimates of differentially encoded phases and was used in the simulation for convenience.

2.4 Approaches to Variable Sequence Size

There are two different approaches to selecting the sequence size N. The first approach consists of setting the sequence size N equal to the burst size b. The phase reference is estimated for the burst duration and the symbol decisions are performed once the whole burst is received. One implication is that, for long burst sizes, the phase reference approximation and the subsequent detection require a delay which is at least as great as the burst size.

The second approach consists of using a sliding window of size N constrained to $N \leq b$. An additional constraint is that each window have the same number of symbols. When $N < b$,

successive windows within the same burst overlap by one symbol; that is, the last symbol of each window becomes the first symbol for the subsequent window, thereby providing a phase reference. Hence, each window has effectively the same number of data bits, that is N - 1. Let the number of windows per burst be a. For a given burst size, the allowed window sizes are only those specified as integers by

$$N = \frac{b - 1 + a}{a}, \qquad 1 \le a \le b-1 . \tag{10}$$

For the ground terminal demodulator, $b = 13$ and the allowed values for N are 2, 3, 4, 5, 7, and 13. Note that the former approach of a variable burst size is indeed a subset of this approach as it corresponds to the case where $b = N$.

One advantage of the window approach is that the system BER performance may be optimized by taking into account frequency offset and trying to adjust accordingly the duration of the phase reference approximation. On the other hand, the burst size, which is often a system specified value, will remain fixed. Another advantage is that the required delay to make the phase reference approximation is not constrained to the whole burst size. Thus, the processing delay may then be reduced for some systems. For SATCOM systems, this delay reduction is not as critical because the demodulator processing delay is generally small compared to the transmission and the payload processing delays.

3 Bit Error Probability Performance

3.1 Theoretical BER Performance

The average bit error probability for MSDPSK with antipodal signals and $N \ge 3$ is found to be approximately upper bounded by [1]

$$P_b \le \frac{2}{\sqrt{\pi\gamma}} \sqrt{\frac{N-1}{N-2}} \left[\frac{1}{2}\exp\{-\gamma\}\right] \tag{11}$$

where γ denotes the bit energy-to-noise spectral density ratio, E_b/N_o. Note that (11) is not valid for $N = 2$ (binary DPSK). For this case, the bit error probability is well known to be

$$P_b = \frac{1}{2} \exp\{-\gamma\} \tag{12}$$

The terms in front of the brackets in (11) correspond to the BER performance improvement factor for MSDPSK over DPSK. The larger N the better the BER in AWGN, though with diminishing returns.

The upper bounds in (11) rely on a sliding window within an infinite burst size. Three different upper bounds for MSDPSK, as determined by (11), are illustrated in Fig. 3a, as a

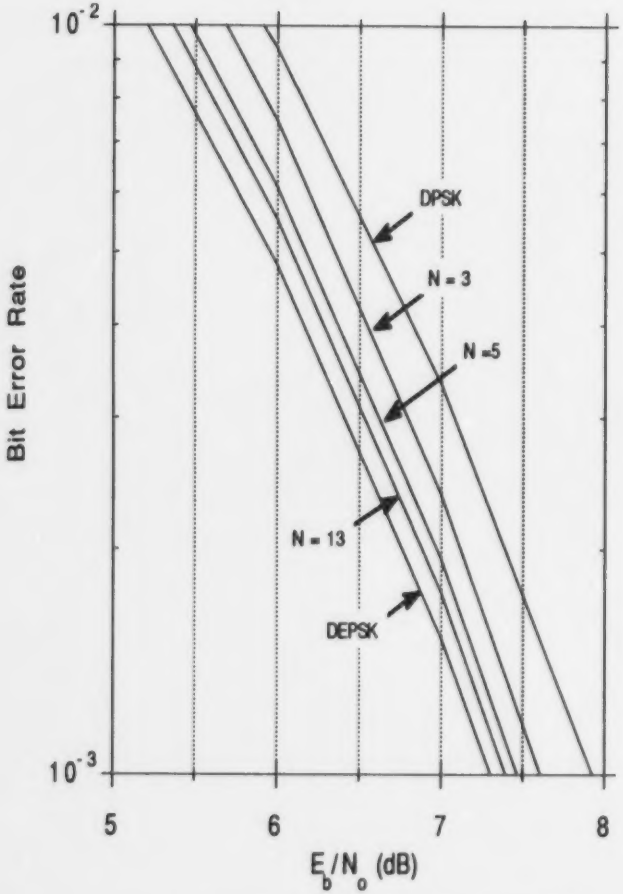


Fig. 3a. BER vs E_b/N_o for the upper bounds of MSDPSK as a function of the window size.

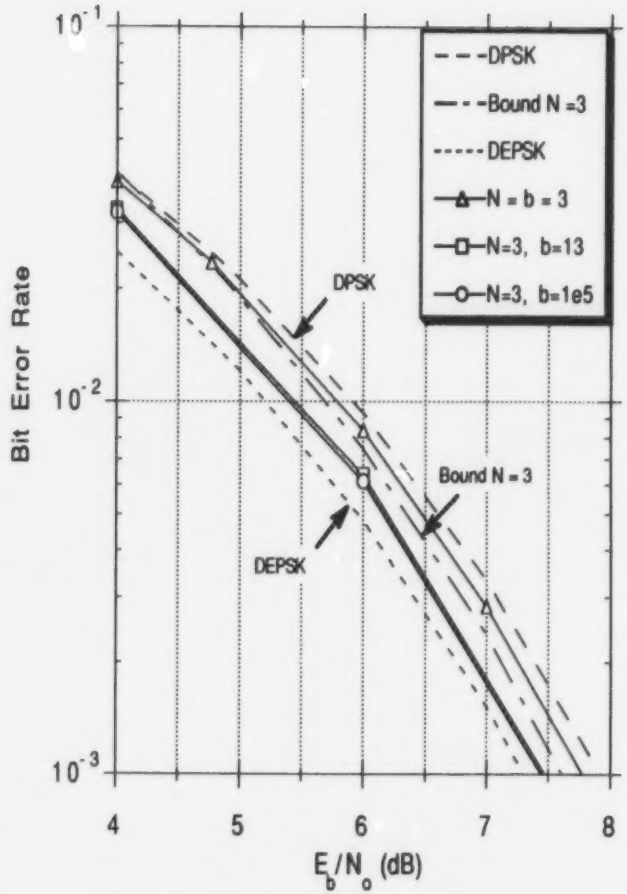


Fig. 3b. BER vs E_b/N_o for N = 3 as a function of the burst size (no frequency offset).

function of the sequence size N. This set of upper bounds for different values of N is itself upper- and lower-bounded. Fig. 3a also illustrates the BER curves of DPSK and DEPSK which constitute the upper and the lower bound, respectively, to MSDPSK. As N increases, the BER of MSDPSK approaches asymptotically the BER of DEPSK. In fact, the majority of BER improvement is achieved for $N = 5$. For $N = 13$, the difference between the upper bound and the DEPSK bit error probability is negligible (<0.1 dB) at a BER of 10^{-3}.

3.2 Simulation Results without Frequency Offset

3.2.1 General

The following section presents simulation results which were obtained for a channel without frequency offset. These results allow a comparison between the upper bounds as specified by (11) and the BER simulation curves. The BER curves were obtained via simulation for $E_b/N_o = 4$, 6, 8, and 10 dB. For these E_b/N_o values, simulations were run until the number of errors, e, reached 10000. According to the central limit theorem, the standard deviation, σ, for these results is $1/\sqrt{e}$. Twenty-four different simulations were performed several times at $E_b/N_o = 6$ dB, and all but one of these simulations had a standard deviation within the expected 1%.

3.2.2 Variable Burst Size

Given a specific window size, the effects of the burst size on the BER performance were investigated. For $N = 3$, Fig. 3b illustrates the upper bound and BER simulation curves for $b = 3$, $b = 13$, and $b = 10^5$. The latter BER simulation curve approximates a continuous-mode transmission for comparison purposes with the upper bound which is based on an infinite burst size ($b = \infty$). Simulation results show that, for a given window size N, the greater b the better the BER performance in AWGN. For a small window size up to 5 bits, the effect is more noticeable. For $b = 13$ and $b = 10^5$, the BER simulation curves are almost identical and both within the upper bound. As shown in Fig. 3b, most of the BER improvement over $b = 3$ is obtained with $b = 13$.

Derivation of the theoretical bit error probability of MSDPSK for burst-mode transmission could confirm these simulation result; however, this derivation was attempted without success. The above simulation results might be explained as follows. Some differences exist between a continuous-mode and a burst-mode implementation. In the continuous mode, the last symbol of a given window is also the first symbol of the following window to preserve the differentially encoding dependency between successive symbols. Only one initial phase reference is required. In a burst mode, a phase reference is required for each burst. The result is that, for a given number of transmitted symbols, less data bits are received for a burst mode than for a continuous mode. As b decreases, the ratio of phase reference symbols to data symbols increases. Noisy phase reference symbols may cause errors yet they are not included in the total number of symbols used for BER calculation.

Fig. 3c illustrates BER simulation curves and their respective upper bounds for no frequency offset as a function of the burst size. For $N = b = 13$, the BER performance approaches that of DEPSK as expected. Clearly, for no frequency uncertainty, the longer N the better the BER performance.

3.2.3 Variable Window Size

Fig. 3d shows the BER simulation curves for no frequency offset and $b = 13$ as a function of the window size. The general behavior of these curves is similar to the one illustrated in Fig. 3c for a variable burst size. As illustrated in Fig. 3d, each BER simulation curve is within its associated upper bound.

3.3 Simulation Results with Frequency Offset

3.3.1 General

Given a fixed amount of frequency uncertainty, Δf, there is an optimum sequence size N that yields the best BER performance. In the presence of frequency offset, only simulation results for $b = 13$ are presented. In this report, the amount of frequency uncertainty is limited by a system specification such that the phase shift, $\delta = 2\pi(\Delta f)T_b$, due to frequency offset at the receiver is no more than 90° for the burst duration. For $b = 13$, a phase shift limitation of 7.20 degrees/bit corresponds to a frequency offset of no more than 1/50 the symbol rate. Hereafter, the amount of frequency offset per bit is specified with its corresponding phase shift per bit, with the units degree/bit shortened to deg/bit.

3.3.2 Variable Window Size

Figs. 4a through 4c each illustrates BER simulation curves corresponding to window sizes of 5, 7, and 13 bits as a function of the phase shift, which ranges from 0 deg/bit to 7.20 deg/bit. For $N = 5$, the phase shift produces a negligible effect on the BER curve; for a 7.20 deg/bit phase shift the BER is within 0.3 dB of the BER curve with no frequency offset at a BER of 10^{-3}. As N increases, however, the BER degradation increases significantly. For $N = b = 13$, the degradation is almost 2 dB at a BER of 10^{-3}. As expected, the larger the window size, the larger the BER performance degradation due to a specific phase shift or frequency offset.

The impact of a specific phase shift on various window sizes is illustrated in Figs. 5a through 5c. For a phase shift of 4.32 deg/bit, all the BER simulation curves but the one for $N = 13$ are very close to each other. When the phase shift is increased to 5.76 deg/bit, the offset is large enough to cause the BER curve of $N = 13$ to get worse than DPSK at a BER of 10^{-2}. For this phase shift, the optimum window size lies in the range of 3 to 7. A further increase of the phase shift to 7.20 deg/bit shows that the optimum window size is now 3. At 7.20 deg/bit, the BER curve for $N = 3$ is the only one that still outperforms DPSK at a BER of 10^{-5}.

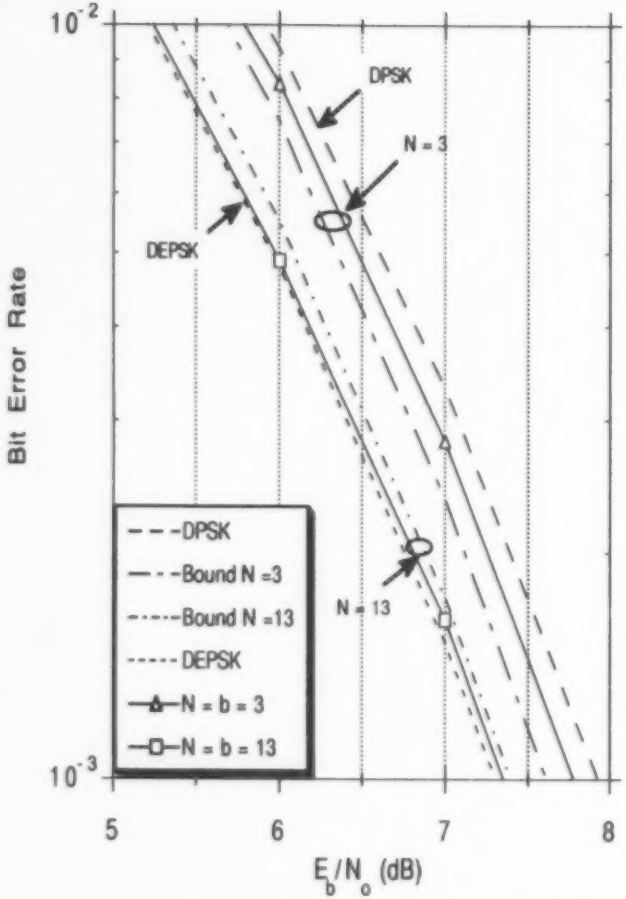


Fig. 3c. BER vs E_b/N_o as a function of the burst size (no frequency offset).

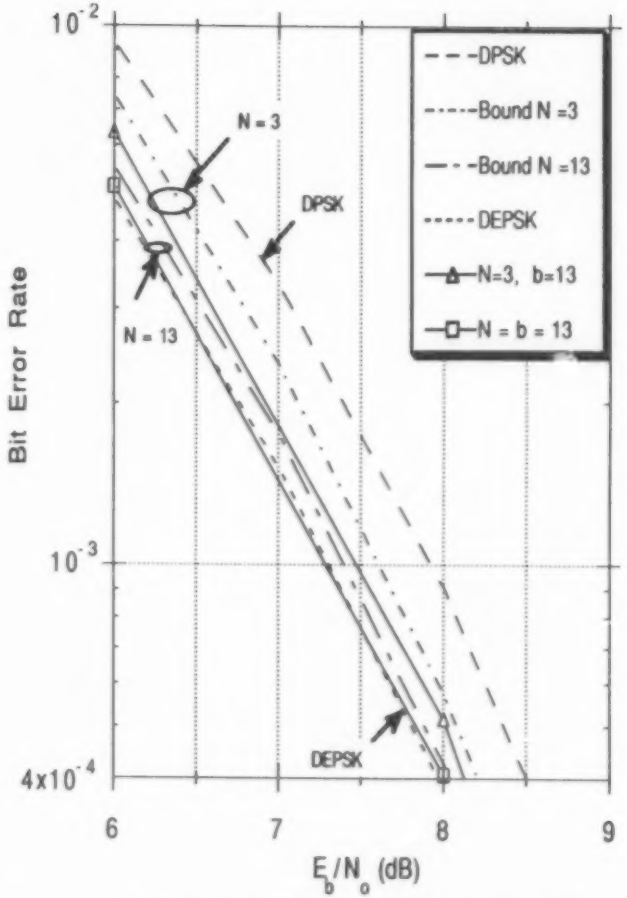


Fig. 3d. BER vs E_b/N_o as a function of the window size (no frequency offset).

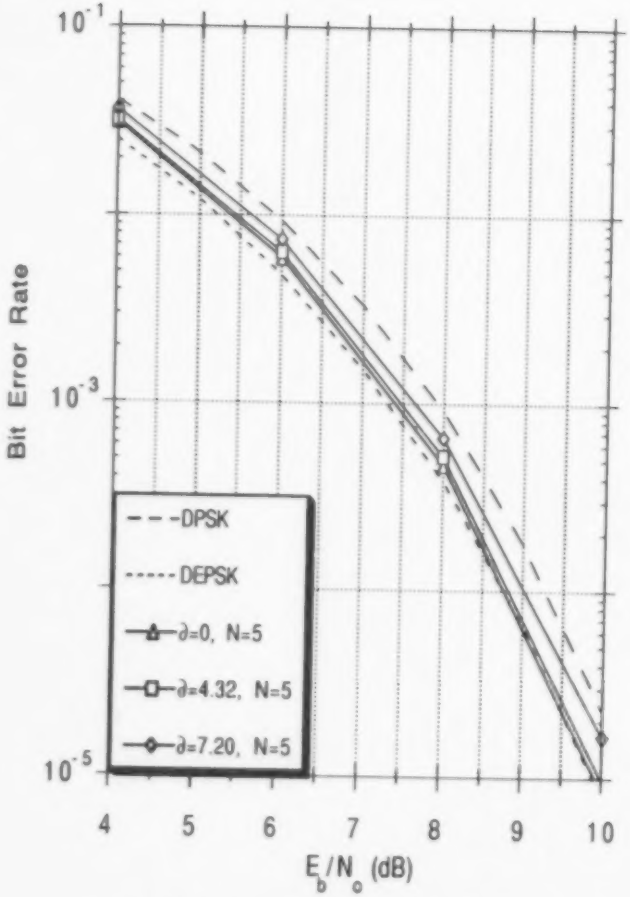


Fig. 4a. BER vs E_b/N_o with b = 13 and N = 5 as a function of ∂ in degree per bit.

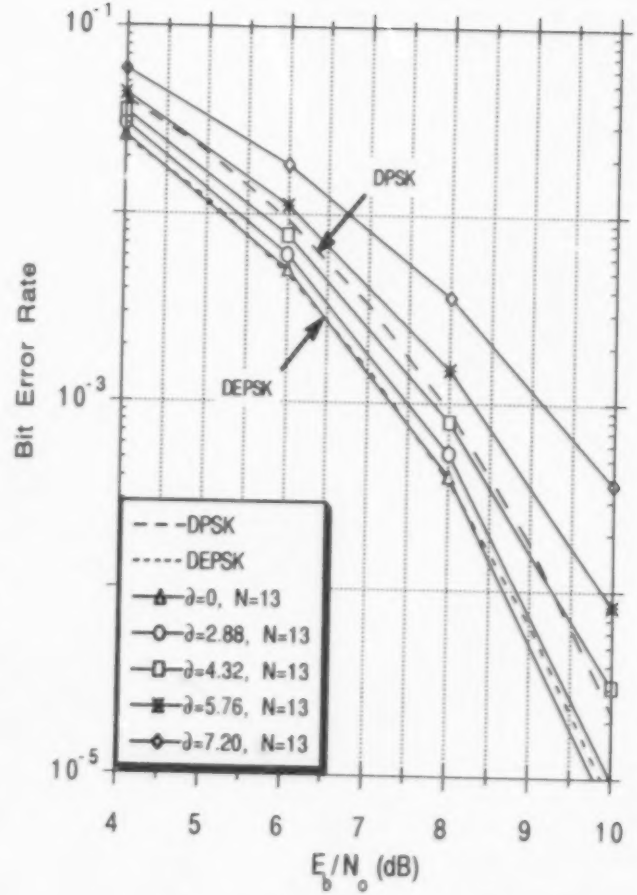


Fig. 4b. BER vs E_b/N_o with b = N = 13 as a function of ∂ in degree per bit.

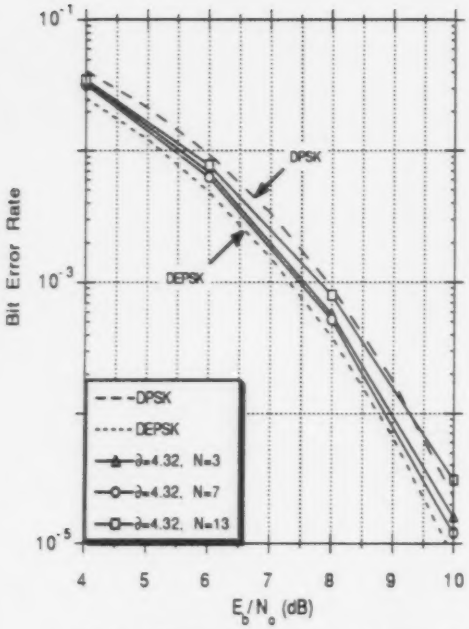


Fig. 5a. BER vs E_b/N_o with ∂ = 4.32 degree/bit and b = 13 as a function of the window size.

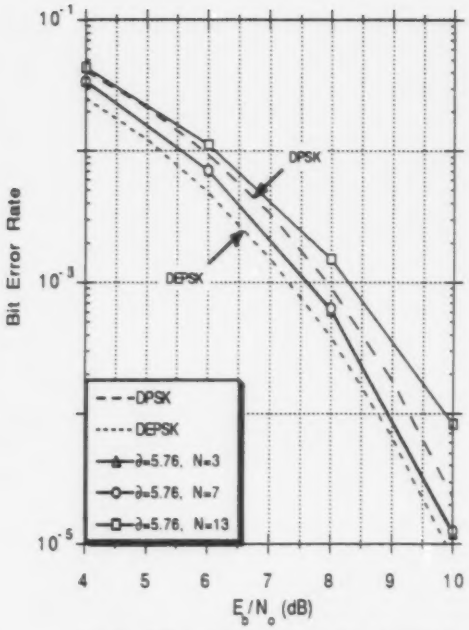


Fig. 5b. BER vs E_b/N_o with ∂ = 5.76 degree/bit and b = 13 as a function of the window size.

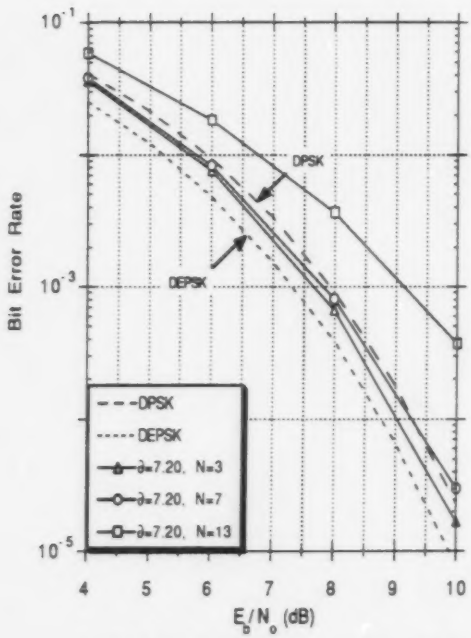


Fig. 5c. BER vs E_b/N_o with ∂ = 7.20 degree/bit and b = 13 as a function of the window size.

3.4 Future Work

Some BER performance could be gained by estimating, with proper processing, the frequency offset effects over the burst duration. This frequency offset estimate could then be removed from the received signal, allowing MSDPSK to be more robust against frequency uncertainty. Consequently, MSDPSK schemes with large sequence sizes would be less affected by frequency offset and would still be attractive when the latter is significant. In [7], studies have shown that, with a frequency-offset correction scheme, DEPSK with a four-phase signal set could withstand phase shift up to almost 45 deg/bit with a negligible degradation. A similar frequency-offset correction scheme for MSDPSK will be presented in a forthcoming paper.

4 Conclusions

The impact of frequency offset on MSDPSK was presented for a burst-mode transmission with a variable sequence or window size for the phase reference approximation. Simulation results indicate that an optimum window size N is associated with a specific frequency offset in terms of BER performance. For a long sequence size, the BER performance degradation due to frequency offset increases very rapidly. For a burst size of 13, the optimum window size for the specified range of frequency offset is 5.

MSDPSK has been shown to outperform DPSK in terms of BER performance. In fact, the MSDPSK scheme has been shown to approach DEPSK BER performance with as few as 13 symbols.

Appendix A: Derivation of Equation (9)

In [1], the decision rule for MSDPSK is

$$\eta = \left| \sum_{i=0}^{N-2} r_{k-i} e^{-j \sum_{m=0}^{N-i-2} \Delta\hat{\phi}_{k-i-m}} + r_{k-N+1} \right|^2 \tag{13}$$
$$= \left| r_k e^{-j(\Delta\hat{\phi}_k + \ldots + \Delta\hat{\phi}_{k-N+2})} + \ldots + r_{k-N+2} e^{-j(\Delta\hat{\phi}_{k-N+2})} + r_{k-N+1} \right|^2$$

where N denotes the sequence size, r_k is the sampled signal in the k^{th} transmission interval, $\Delta\phi_k$ is the input symbol phase, and $\hat{}$ denotes estimates at the demodulator. Similar to (3), the k^{th} differentially encoded phase is given by

$$\hat{\phi}_k = \hat{\phi}_{k-1} + \Delta\hat{\phi}_k \tag{14}$$

With (14), one may write

$$\hat{\phi}_{k-i} - \hat{\phi}_{k-N+1} = \sum_{m=0}^{N-i-2} \Delta\hat{\phi}_{k-i-m} \tag{15}$$

Combining (13) and (15), the decision rule statistic becomes

$$\eta = \left| r_k e^{-j(\hat{\phi}_k - \hat{\phi}_{k-N+1})} + \ldots + r_{k-N+2} e^{-j(\hat{\phi}_{k-N+2} - \hat{\phi}_{k-N+1})} + r_{k-N+1} \right|^2 \tag{16}$$

Since the decision rule statistic is the magnitude of a vector summation, the addition of an arbitrary fixed phase, ϕ_a, to each vector in (16) results in the same decision. Its effect is simply to rotate by that angle the two sequence vectors with maximum magnitude. Thus, (16) is equivalent to

$$\eta = \left| r_k e^{j(\hat{\phi}_k - \hat{\phi}_{k-N+1} + \phi_a)} + \ldots + r_{k-N+2} e^{j(\hat{\phi}_{k-N+2} - \hat{\phi}_{k-N+1} + \phi_a)} + r_{k-N+1} e^{j\phi_a} \right|^2 \tag{17}$$

If we let $\phi_a = \phi_{k-N+1}$, the phase of the first sequence signal, then (17) is simply

$$\eta = \left| r_k e^{j\hat{\phi}_k} + \ldots + r_{k-N+2} e^{j\hat{\phi}_{k-N+2}} + r_{k-N+1} e^{j\hat{\phi}_{k-N+1}} \right|^2 \tag{18}$$

The decision rule in (18) relies on estimates of differentially encoded phases whereas estimates of input symbol phases are used in [1]. Both decision rules are optimum but (18) was used in the simulation for convenience.

Appendix B: SPW System Implementation

B.1 General

This appendix describes the MSDPSK SPW system implementation. The system model is the top design level and is composed of library and user-defined symbol models. The user-defined symbol models are presented with their associated detail models. The SPW library symbol models may be found in [10]. The aim of Appendix B is to show typical implementation details that any SPW user has to address in designing SPW systems. Since a system implementation is not unique, the user-defined detail models are described only briefly. The main system parameters are also discussed, as well as some other implementation details.

B.2 Block Implementation

This section briefly describes the function of each user-defined block of the complete MSDPSK low-pass equivalent system which is shown in Fig. 6. The SPB random data block generates a random stream of bits. These bits are differentially encoded and phase modulated as shown in Figs. 7 and 8, respectively. The signal is then transmitted through a channel with AWGN as detailed in Fig. 9 [9]. The channel can exhibit Rayleigh fading but the latter remained disabled for the simulations. Next, some frequency offset is added to the signal with the SPB complex spectral shifter (the block is removed for simulations without frequency offset). A random phase, constant for each burst, is then added to the signal as per Fig. 10.

The demodulator consists of the remaining blocks. The block *DEPSK ref_win*, shown in Fig. 11, calculates the phase reference for each window. This block is the most complex in the whole system, mainly because of the timing control requirements. Several switches are used to timely reset the phase reference estimation given the window and burst sizes. There is also several feedback loops for continuous summation. The phase reference estimate is then compared to the phase of each symbol in *DEPSK Ref_Comp*, as detailed in Fig. 12. Next, the block *set_bit* (see Fig. 13a) forces the phase reference, the first decoded bit of each burst, to be 0. Similarly, the block *set_bit_del* (see Fig. 13b) forces the first data bit of each burst to be 0. The data bit stream is also delayed for proper synchronisation with the decoded bit stream. Finally, the block *iteration_cntr* calculates the system bit error rate (BER) as per Fig. 14. The hold signal on the signal sink block ensures that only one BER value is stored, thus keeping signal files to a minimum in terms of memory size.

B.3 Implementation Details

B.3.1 Main System Parameters

The main simulation parameters are shown in Fig. 6. The burst size b is only constrained by $b \geq 3$ while the window size N is constrained by (10). Hence, the MSDPSK system

Appendix B: SPW System Implementation

"LOW-PASS EQUIVALENT FOR A DEPSK SYSTEM IN AWGN"

"FILE: mlib/depsk_sys.system"

SIMULATION PARAMETERS

MAIN PARAMETERS:

		PARAMETER NAME:
Sampling freq:	1.0 (Hz)	(s_freq)
Frequency offset:	0.02 (Hz)	(offset)
Eb/No:	02.512 (linear)	(Eb/No)
Bit rate	1.0 (Hz)	(bit_rate)
Clock interval	1.0 (samples/bit)	(interval)
Burst size	13.0 (bits/burst)	(del)
Error number	10000.0 (errors/test)	(errors)
Window size	5.0 (bits)	(window)

By default, the ouput is written to test_lib/sig01 in ascii

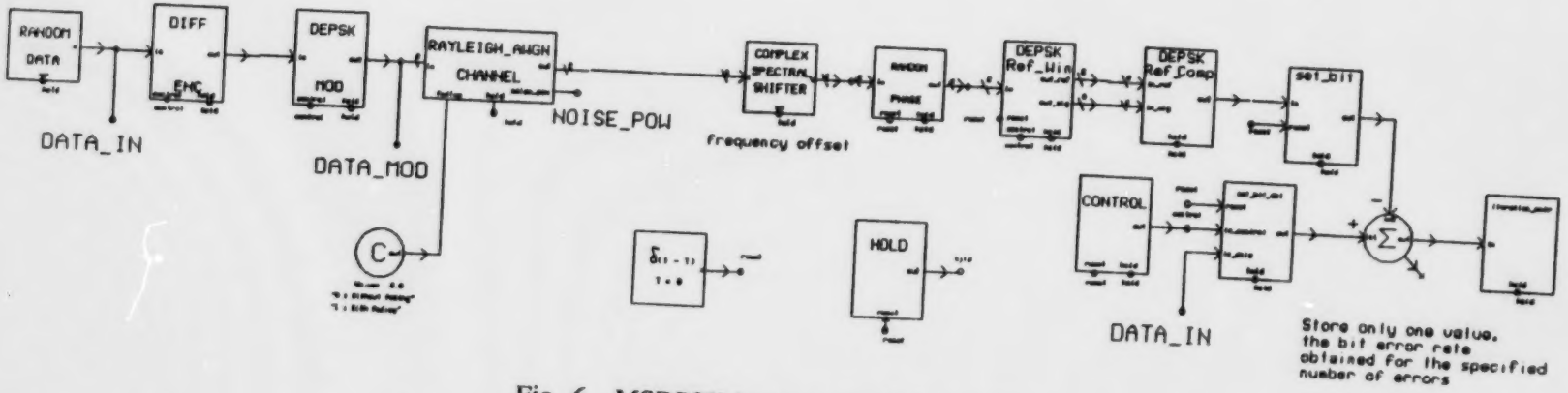


Fig. 6. MSDPSK low-pass equivalent system.

"FILE: mlib/depsk_diff_enc.detail"

DIFFERENTIAL ENCODING BLOCK PARAMETERS		
MAIN PARAMETERS:		
Initial value	0.0	(hold_in_val)

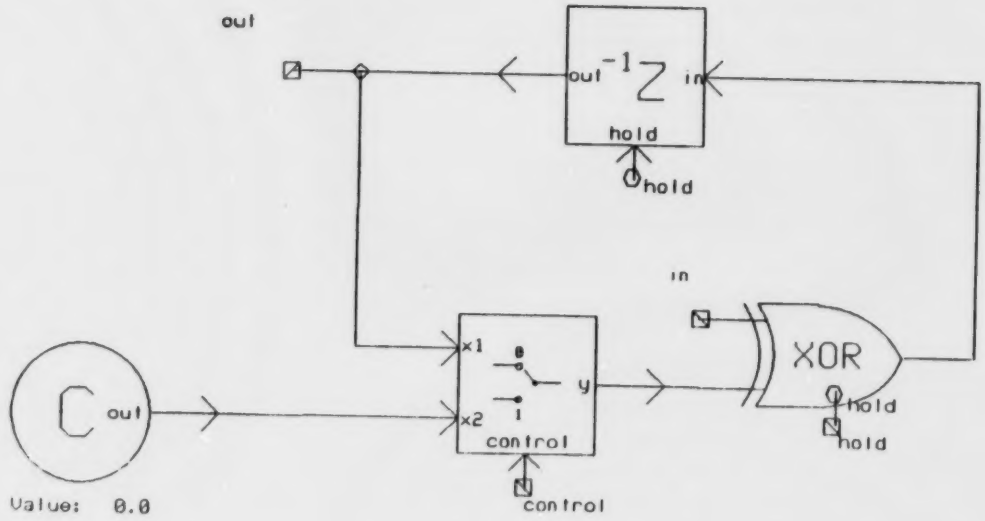


Fig. 7. Differential encoder.

Appendix B: SPW System Implementation

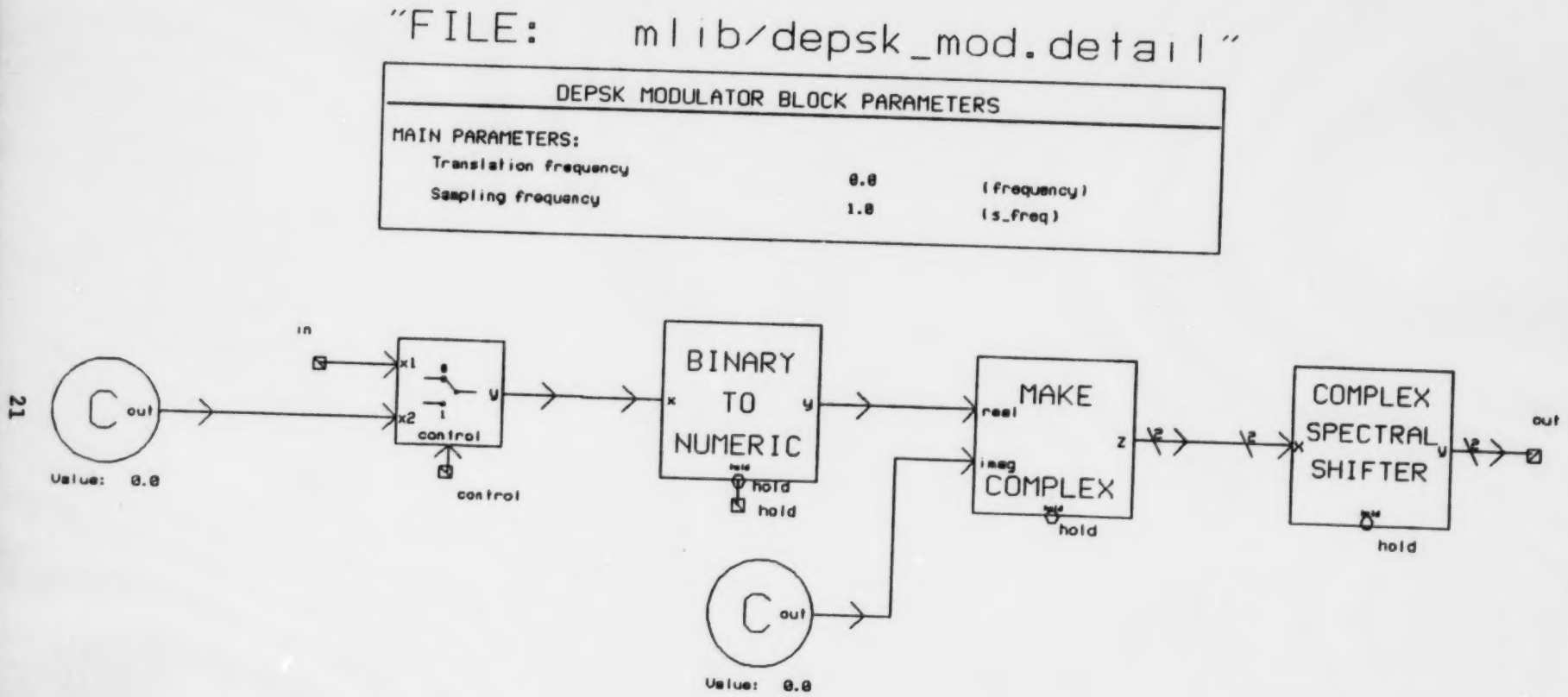
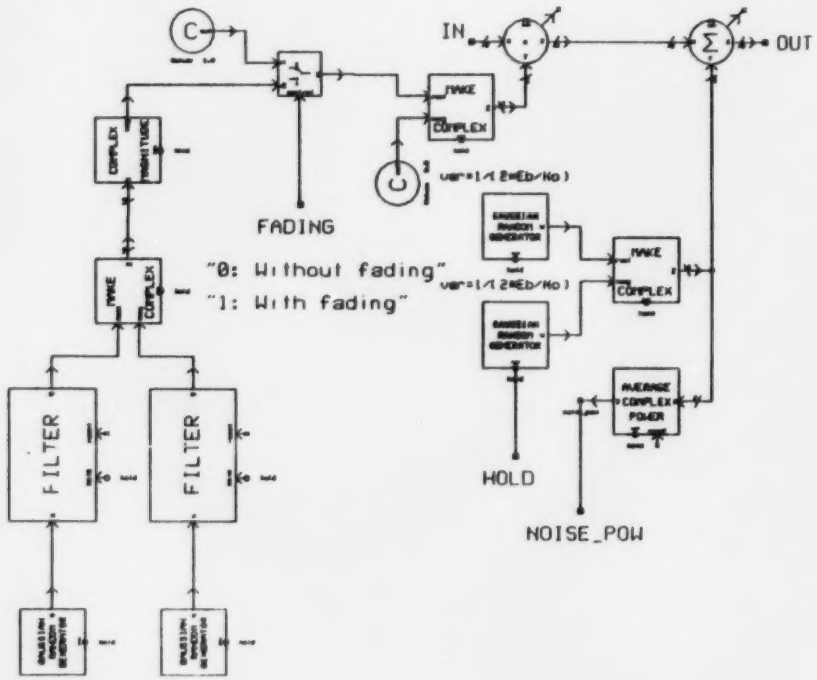


Fig. 8. Phase modulator.

Appendix B: SPW System Implementation

SIMULATION PARAMETERS			
MAIN PARAMETERS:			
Sampling freq:	1.0	(Hz)	(s_freq)
Translation frequency:	0.0	(Hz)	(frequency)
Eb/No:	02.512	(linear)	(Eb/No)



FILE: mlib/channel.detail

Fig. 9. Channel with Rayleigh fading (optional) and AWGN.

Appendix B: SPW System Implementation

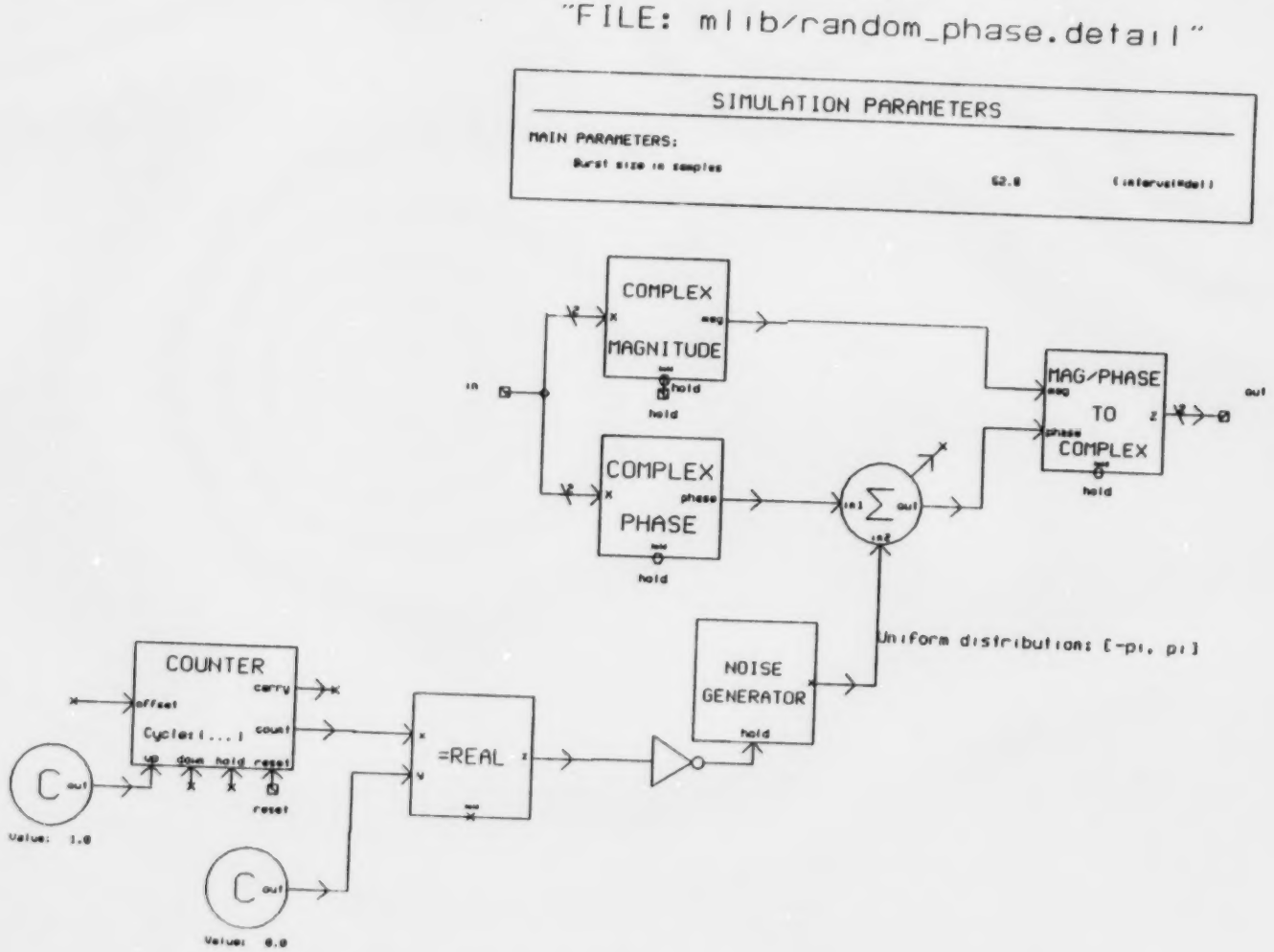


Fig. 10. Random phase channel effect for each burst.

Appendix B; SPW System Implementation

"FILE: mlib/depsk_ref_win.detail"

DEPSK REFERENCE CALCULATION BLOCK PARAMETERS

MAIN PARAMETERS

Clock interval	20.0	(interval)
Burst size	13.0	(del)
Window size	5.0	(window)

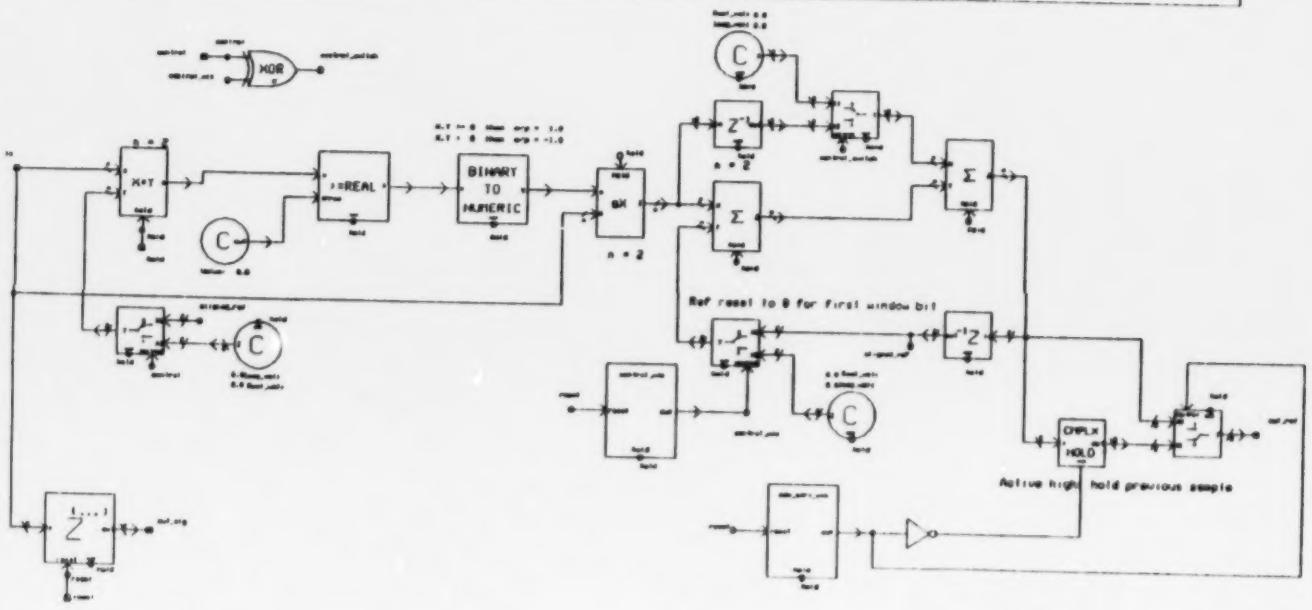


Fig. 11. Phase reference estimator for each window.

Appendix B: SPW System Implementation

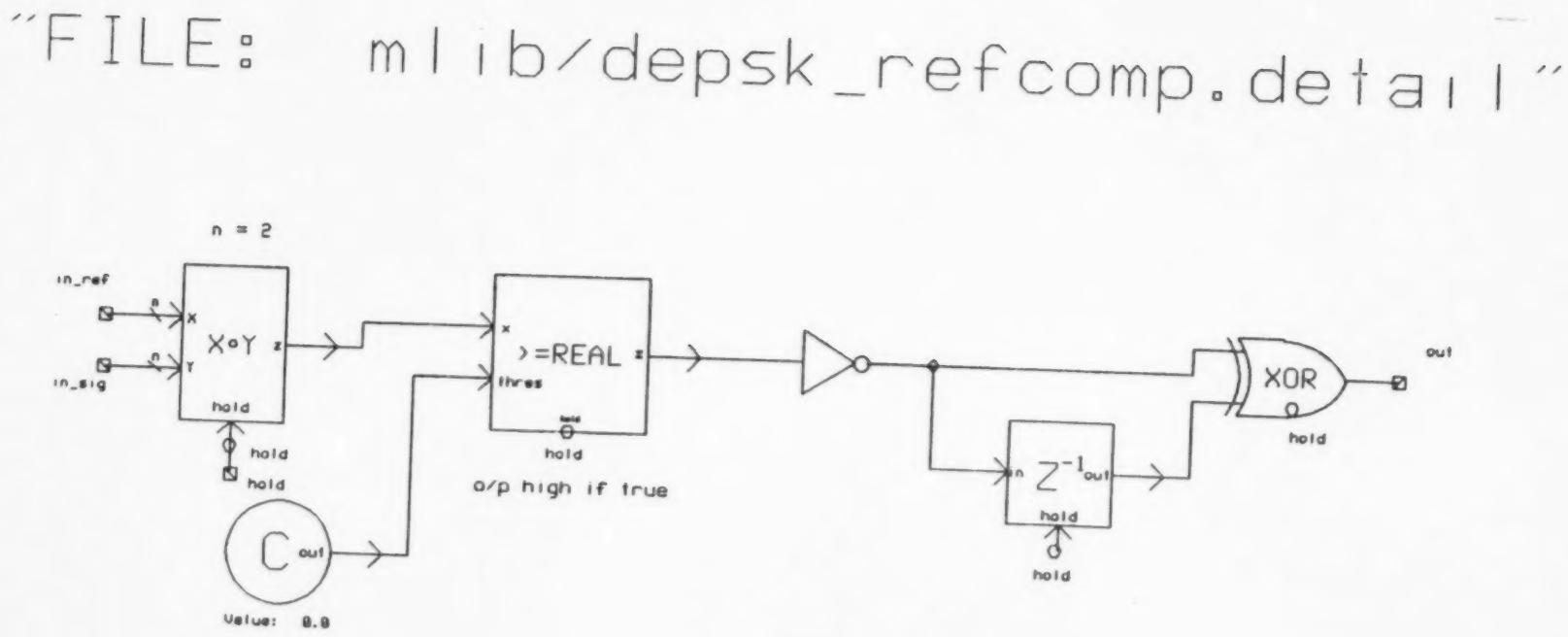


Fig. 12. Symbol phase and phase reference comparator.

Appendix B: SPW System Implementation

"FILE: mlib/set_bit.detail"

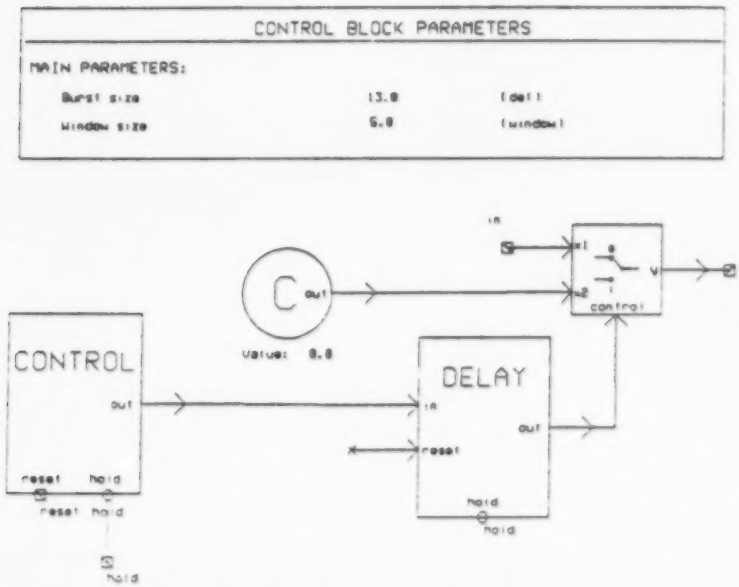


Fig. 13a. Bit reference adjustment for decoded bit stream.

"FILE: mlib/set_bit_del.detail"

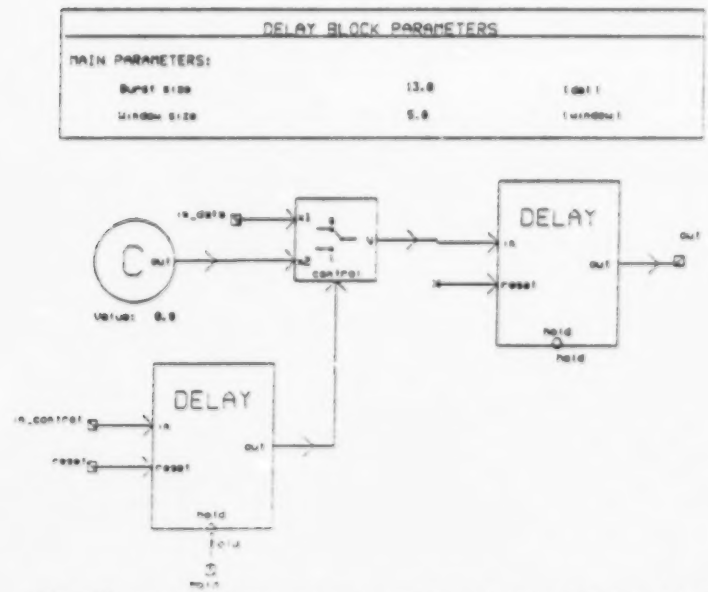


Fig. 13b. Bit reference adjustment and synchronisation of data bit stream.

Appendix B: SPW System Implementation

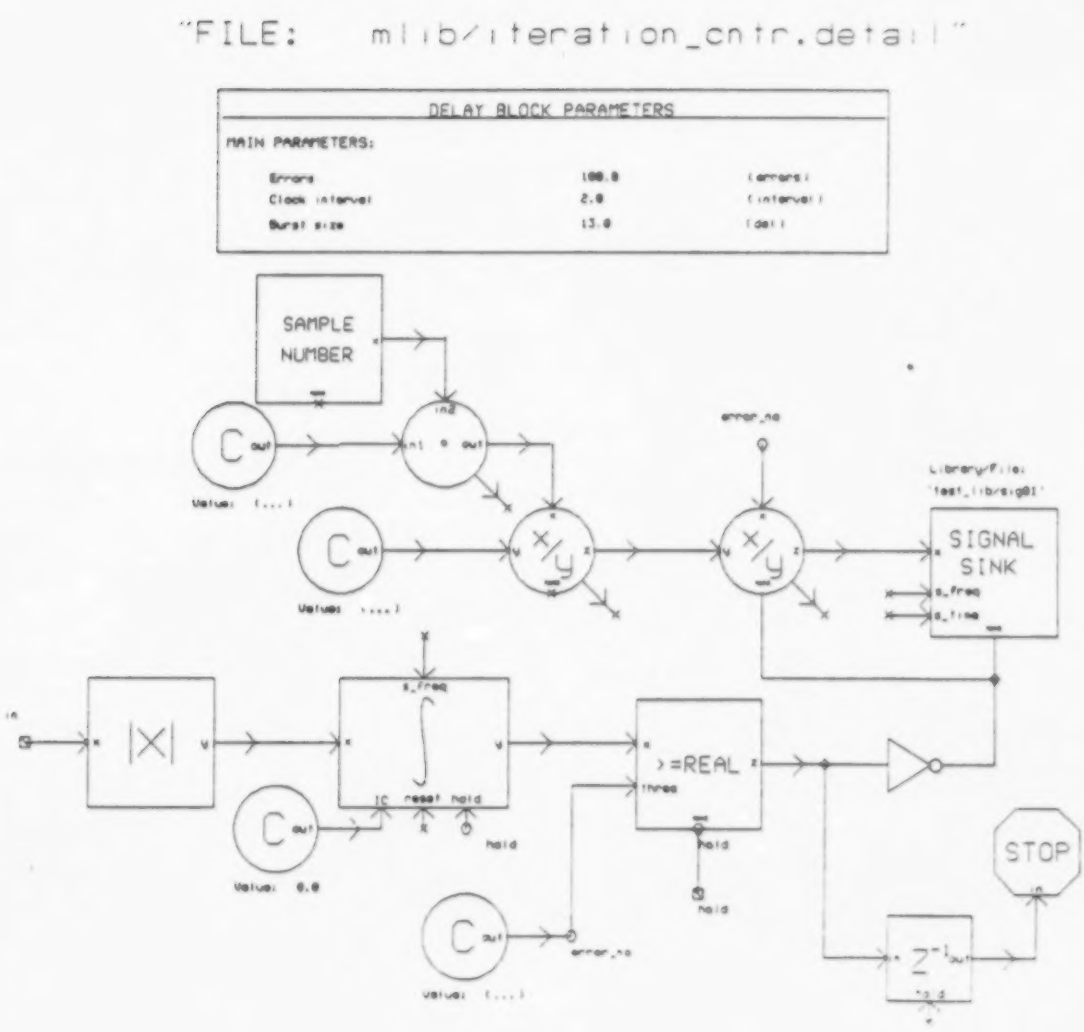


Fig. 14. Bit error rate calculator.

implementation cannot simulate DPSK ($b = 2$). The error number parameter was set at 10000. This number yielded a relatively low variance at the expense of much longer simulations for larger values of E_b/N_o. The bit rate parameter is a fixed system specification and is normalized to one in the SPW system implementation. The frequency offset parameter corresponds to Δf as described in Section 3.3.1.

The last two simulation parameters are clock interval and sampling frequency. Their purpose is better explained in light of the fact that some provision was originally made to analyze the effects of timing uncertainty on MSDPSK by decimating the samples of the signal. The clock interval parameter may be used for this decimation through the use of the hold signal. The advantage of this approach is that the hold signal is common and that decimation is easily implemented. Basically, each symbol is held for a duration equal to the clock interval value. With frequency uncertainty only, the clock interval value is fixed at one. Any other interval would yield identical results but would increase simulation time by a factor proportional to the interval value. The sampling frequency parameter was fixed at one. In SPW, this parameter seems to be used mainly for the signal representation in the Signal Diagram Editor (SDE) as it is mainly provided with signal generation/storage blocks. The parameter does not prevent other system blocks to operate on a sample by sample basis.

B.3.2 Hold

The hold signal detailed in Fig. 15 may have different purposes. First, the hold signal may be used to delay a timing control signal such as an enable. Second, it may be tied to each block having such a parameter to temporarily hold an on-line simulation. (An on-line simulation allows result analysis in SDE while the simulation is running.) Third, the hold signal may be used to decimate the samples of a signal. A combination of these purposes was chosen for the MSDPSK system implementation to make the blocks as general as possible. In general, a hold signal is used on each block having such a parameter unless a hold signal would compromise the sample decimation. Hold signals were even provided to blocks that did not affect decimation integrity (e.g. exclusive-or blocks). The only difference is an operation on the first sample of each symbol rather than all samples. For many blocks, processing the first sample and holding the result for a symbol duration is equivalent to processing all samples since the latter have the same value within a symbol interval.

B.3.3 Clocks

Figs. 17a-17c illustrate three detail models of master clocks which are used in the MSDPSK system implementation. Fig. 17a illustrates the first clock, control, which is set for the first bit of each block. Fig. 17b shows the second clock, control_win, which is set for the first bit of the first window of each burst and for the second bit of the other windows. Finally, the clock adv_ctrl_win is set for the last bit of each window and is shown in Fig. 17c.

Appendix B: SPW System Implementation

"FILE: mlib/hold.detail"

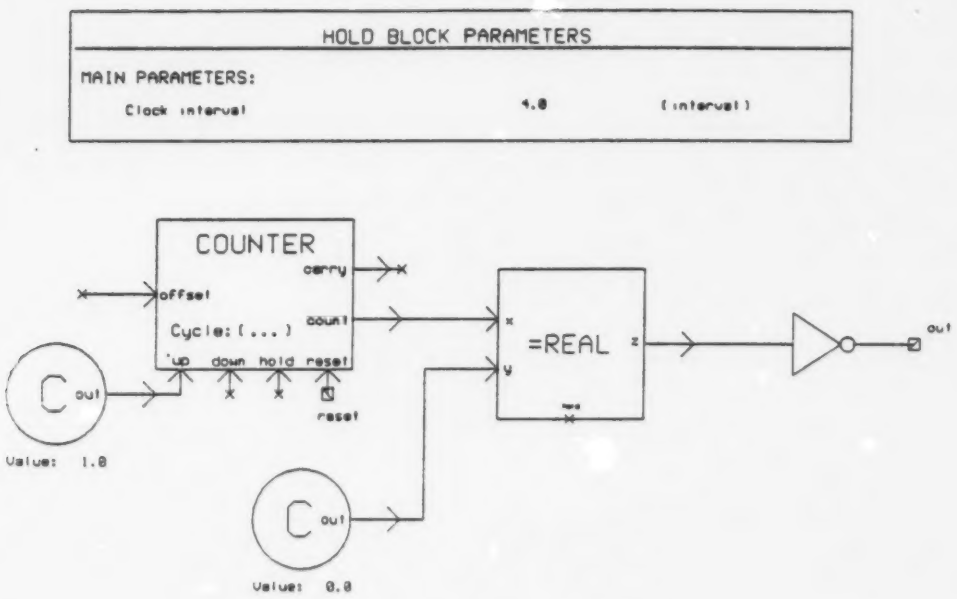


Fig. 15. Detail block of hold signal.

"FILE: mlib/delay.detail"

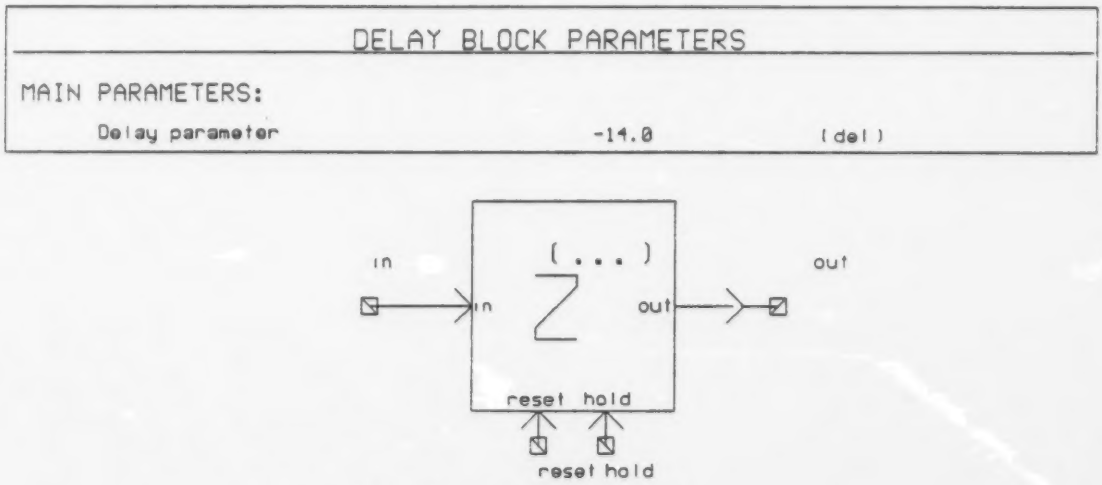


Fig. 16. Detail block of delay.

Appendix B: SPW System Implementation

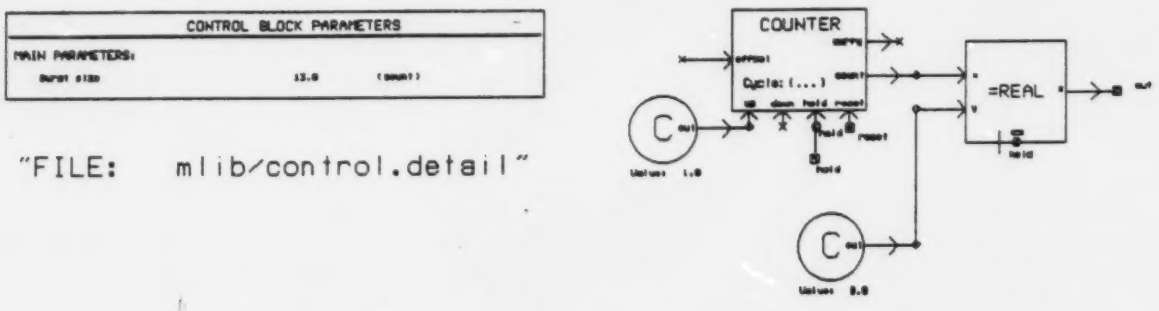


Fig. 17a. Timing control specified by the start of each burst.

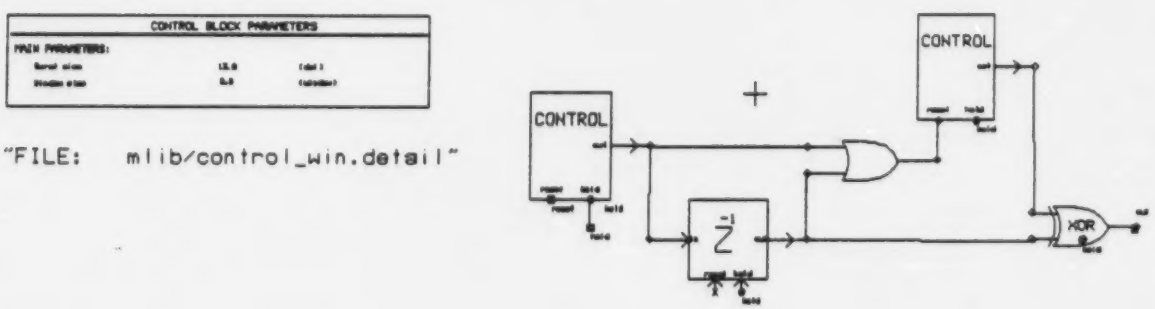


Fig. 17b. Timing control specified by the start of each window.

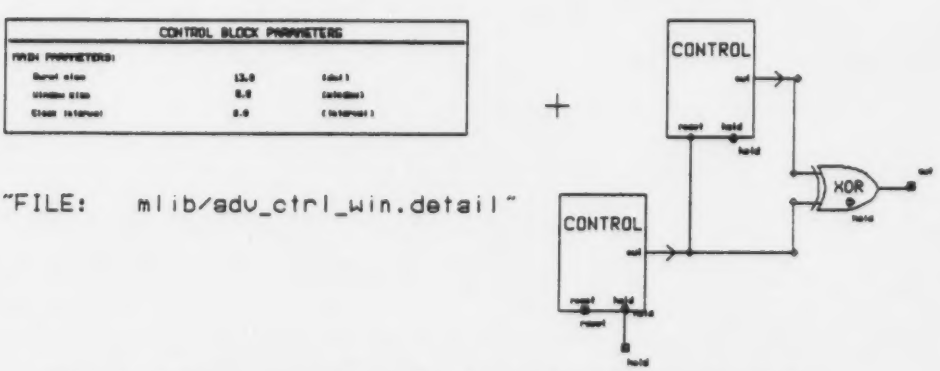


Fig. 17c. Timing control specified by the last bit of each window.

B.3.4 Reset

A typical SPW counter output for a cycle of 4 for example is 1, 2, 3, 0, 1, 2, 3, 0 and so on. A reset signal corresponding to a unit impulse is used to make the counters start their first sequence at 0. For some counters, the reset signal was intentionally left out to get the original output.

For delay instances, connections are not made to the reset signal. The reason is that the input value at the time of the reset is lost while the other values are properly delayed. The same concept applies to *mlib/delay.detail* (see Fig. 16) even though a reset port was included to make this user-defined block as general as possible at the lowest design level. In fact, the reset was generally included as a port in several hierarchical levels; however, the final connection is never made to the reset signal.

Appendix C: Batch Execution

C.1 General

Batch simulations can save a significant amount of time to a SPW user. There is no user interaction necessary and simulations may be run after working hours. This Appendix presents two approaches to perform batch SPW simulations. The main difference between the two approaches is their intended use for specific degrees of system completion. The first approach corresponds to batch simulations based on the SPW system model, thus making this approach very convenient for a system still at the design level. The second approach consists of batch simulations which rely on a modified version of the *C* program generated by the Code Generation System (CGS). The main advantage of the second approach is that the use of a *C* program allows the exit of the *X* window environment without affecting batch SPW simulations. Thus, the workstation does not have to be dedicated to SPW and the workstation multiprocessing capabilities may be fully used.

C.2 Batch Simulations at the Design Level

Simulations at the design level are very useful when performing the final tests to obtain the final system implementation. This section presents a step-by-step procedure to obtained batch SPW simulations based on *userlib/system_name.system*. The procedure is based on the BDE macro language [10]. The idea consists of recording a macro for one complete simulation execution of a system for which the netlist has been successfully generated and evaluated. Then, changes are made to the macro language file to execute multiple simulation without user interaction. An example of such a batch file macro is given in Section C.4.

1. In BDE, start recording a macro (e.g. *userlib/batch*) and perform the following commands and procedures:

   a. **ADD VIEWPORT** (make viewport as large as possible);

   b. **DISPLAY VIEW** ***userlib/system_name.system*** **EDIT**;

   c. This step defines the exact location of each parameter value. For each simulation parameter, point to the end of the parameter value and backspace until the value is completely erased. Type in another possible value. Do this step for all signal sink filenames (use numbered filenames, e.g. *sig100.sig*) for which the simulation results need to be saved;

   d. **ANALYSIS GENERATE SELECTXY tool=cgs**;

   e. **ANALYSIS EVALUATE COMPLETE CGS_run GEN_PGM**

COMPILE_PGM RUN_PGM tooldeck=*userlib/system_name.system*;

f. **SYSTEM RECORD STOP.**

2. Use the command **SYSTEM REPLAY *userlib/batch*** to verify a single execution. Repeat Steps 1 and 2 until successful.

3. Using a text editor (e.g. vi), edit *userlib/batch* according to the following general comments:

   a. An important concept is understanding how an argument is passed to a parameter. Briefly, each simulation parameter is associated to a string of characters. Each string is made of backspaces (to erase the previous value) and its current value. The ENTER statement then inserts this string into the selected viewport. Therefore, to declare variables, use either a *real* or *integer* variable for each simulation parameter, with an associated *xstring* variable.

   b. To allow modification of simulation parameters, use nested *while* loops for each simulation parameter;

   c. Account for *noise_seed* to be modified for each simulation.

4. With only a few number of short simulations, use the command **SYSTEM REPLAY *userlib/batch*** to verify the macro. Go back to Step 3 if necessary.

5. Before each execution of the macro, change *noise_seed*. Change loop and variable values if desired.

C.3 Batch Simulations for a Completed System

This section presents a step-by-step procedure to obtained batch SPW simulations based on a *C* program.

1. If not already available, obtained simulation results in SPW for a specific set of parameter values. Set the storage format parameter in the signal store blocks so that signals are stored in ASCII.

2. In the system model, assign different values to all the simulation parameters that the user wants to change for different simulations. At this point of time, values do not have to be restricted to system specifitions. All the assigned values must be very different from each other (e.g. 22, 333).

3. Use the command **ANALYSIS GENERATE tool=cgs**. Make corrections to the system model, if necessary, and repeat until successful.

4. Use the command **ANALYSIS EVALUATE CGS_RUN GEN_PGM COMPILE_PGM RUN_PGM iterations=333 noise_seed=555**. Make corrections to the system model, if necessary, and repeat Steps 3 and 4 until successful.

5. In the UNIX operating system, use the command **cd /caedata/*userlib*/*system_name*/cae*/\$gencode/cgs** to change directory to where the generated SPW *C* program is located. A file listing would show the *C* program with the name *system_name.c*.

6. Rename *system_name.c* with a name such as *run.c*. This name change is necessary for the following steps but it is good practice to prevent accidental overwrites by future ANALYSIS EVALUATE commands.

7. Repeat Steps 2 to 6 for different simulation parameter values. Do not modify the parameter values which will remain fixed throughout all the simulations. Use a different filename for the new generated program (e.g. *run2.c*).

8. Use the commands **diff *run.c run2.c* > *listing*** and **lpr *listing***. The filename *listing* contains all the variable names with values which are derived from at least one of the simulation parameter values in Step 2. In general, a few variables are associated with each parameter. For each variable in *listing*, derive its value from the simulation parameters. For example, the variable P_del_620_zdel could be associated in *run.c* and *run2.c* with values -21 and -43, respectively. This could mean that P_del_620_zdel = -1 * (Simulation_Parameter - 1), given that Simulation_Parameter was assigned the values 22 and 44 in Step 2.

9. Using a text editor (e.g. vi), locate the section *spwGetArgs* in *run.c*. All basic program modifications can and should be restricted to the *switch* statement that follows the *spwGetArgs* section. For example, the software necessary to pass the iteration parameter as an argument at run time is as follows:

```
case 'i':
   if (i+1) < argc
         spwmax = atol(argv[i+1];
   else {
         printf("usage: system_name [-i iterations] [-n noise_seed]\n");
         spwExit(SYS_FATAL);
```

```
} /*else*/
break;
```

where 'i' is the case label associated to the simulation parameter.

For each simulation parameter, assign a case label and add a case statement to define the variables that were found in Step 8 to be derived from that simulation parameter. Let a simulation parameter called *burst size* be associated with the case label *b*. The additional software required in *run.c* would be as follows:

```
case 'b':
   if (i+1) < argc {
         P_count_804_counter = P_count_1960_counter = atol(argv[i+1]);
         P_del_620_zdel = -1 * (P_count_1960_counter - 1);
         P_const_val_4060_const = (double) P_count_1960_counter - 1.0;

   else {
         printf("usage: system_name [-i iterations] [-n noise_seed]\n");
         spwExit(SYS_FATAL);
   } /*else*/
   break;
```

where all the variables (e.g. P_count_804_counter) would be obtained from *listing*. In this example, the variables are derived from only one simulation parameter. Should a variable be derived from two or more simulation parameters, then the value of this variable must be assigned after both simulation parameters have been passed as arguments to other variables. Moreover, the type specifier (e.g. long, double) of each *C* variable must be respected. For instance, a *long* argument value is passed to a *long* variable with *atol* while *atof* is used for *double* variables. The correspondence between variables with different type specifiers may be forced such as in the third expression assignment above.

10. At the very end of the program, it is useful to print the variable values. It ensures that argument values are passed correctly and it shows which parameters are used for the current simulation. An example of such a command is as follows:

```
printf("system_name: using \n iterations = %ld\n noise_seed = %ld\n Eb/No = %e
(dB) or SNR = %e\n frequency_offset = %e\n burst_size = %ld\n errors %e\n",
spwmax, spw_noise_seed, 10*log10(1 / P_variance_3156_gauss_ran_gen * 0.5),
1/P_variance_3156_gauss_ran_gen, P_frequency_2664_cmplx_tone,
P_count_1960_counter, P_const_val_4216_const);
```

Appendix C: Batch Execution

11. After saving *run.c*, compile *run.c* in UNIX with **cc -o *run* -I/home/spwsys/include *run.c* /home/spwsys/lib/cgs.a -lm**. Make corrections to *run.c* if necessary and repeat Steps 9 to 12 until compilation is successful.

12. In UNIX, execute *run.c* with the parameter values of Step 1 passed as arguments. Use a command such as **run -i 10000 -n 534 -f 0.01**. Each simulation parameter has a default value. To set a simulation parameter, an hyphen and its case label from Step 9 is used, followed by the desired parameter value. Respect the type specifier for each simulation parameter. Repeat Steps 8 to 12 until the simulation results are identical to the ones obtained in Step 1.

13. Create a *C* program (e.g. *batchgen.c*) that will generate a batch file (e.g *batch.csh*). The bacth file may then be executed in the background by using the command ***batch*** **&**. A batch file for two simulations could be as follows:

    time /home/caedata/*userlib*/*system_name*/cae*/\$gencode/cgs/*run* -i 2000000000 -n 8753498 -x 10 -f 0.0 -b 100000 -w 13 -v 1 -e 1000
    grep '\.' /home/spwdata/*signal_lib*/*sig01.ascsig* > > ~/*temp.sim*
    time /home/caedata/*userlib*/*system_name*/cae*/\$gencode/cgs/*run* -i 2000000000 -n 3333333 -x 8 -f 0.0 -b 100000 -w 13 -v 1 -e 1000
    grep '\.' /home/spwdata/*signal_lib*/*sig01.ascsig* > > ~/*temp.sim*

    where the signal location *signal_lib*/*sig01.ascsig* is as specified in the system model and *temp.sim* is a file which stores the simulation results because *sig01.ascsig* is constantly overwritten.

14. If desired, write C programs for additional processing of simulation results.

C.4 Example of a Batch File Macro at the Design Level

```
------------------------------------------------------------------
-- VERSION BDE-2.7.0
------------------------------------------------------------------
--
-- Record Startup: Wed Jul 10 09:37:46 1991
-- Filename: /home/turgeon/batch
-- Purpose: Macro for batch simulations
--
------------------------------------------------------------------
PROGRAM package_name
--
--VARIABLE DECLARATIONS
--
freq_offset,snr,window_size,file_name:xstring;
--
freq_offset_no,file_name_no:integer;
--
i,j,k,l:integer;
--
snr_no,window_size_no:real;
--
iterations,noise_seed:integer;
--
--
PROCEDURE procedure_name BEGIN
execute("ADD VIEWPORT");
affinity("Instance");
--*****************************************************************
--
noise_seed := 74943993;
file_name_no := 100;
iterations := 200000000;
j := 1;
while (j <= 4) do
--
if (j = 1) then
snr_no := 05.00;
else
if (j = 2) then
snr_no := 07.96;
else
if (j = 3) then
snr_no := 12.62;
else
snr_no := 20.00;
```

```
end if;
end if;
end if;
snr := "\b\b\b\b\b"&xstring(snr_no)&"";
--
l := 1;
freq_offset_no := -20;
while (l <= 6) do
--
freq_offset_no := freq_offset_no + 20;
--
if (l = 6) then
freq_offset := "\b\b\b"&xstring(freq_offset_no)&"";
else
freq_offset := "\b\b"&xstring(freq_offset_no)&"";
end if;
--
k := 1;
while (k <= 4) do
--
if (k = 1) then
window_size_no := 13.0;
else
if (k = 2) then
window_size_no := 7.0;
else
if (k = 3) then
window_size_no := 5.0;
else
window_size_no := 3.0;
end if;
end if;
end if;
--
window_size := "\b\b\b\b"&xstring(window_size_no)&"";
--
i := 1;
while (i <= 2) do
--
execute("ADD VIEWPORT");
selectxy(135,841,1143,262);
execute("DISPLAY VIEW model=mlib/depsk_sys.system EDIT");
selectxy(234,356,234,356);
--
noise_seed := noise_seed + 23;
file_name_no := file_name_no + 1;
file_name := "\b\b"&xstring(file_name_no)&"";
--
```

```
enter(""&freq_offset&"",650,694);
enter(""&snr&"",643,673);
enter(""&window_size&"",628,576);
--
execute("DISPLAY PUSH NEXT");
selectxy(1040,405,1040,404);
execute("DISPLAY ZOOM REGION");
selectxy(790,619,1031,384);
execute("DISPLAY PUSH NEXT");
selectxy(1040,405,1040,404);
execute("DISPLAY ZOOM REGION");
selectxy(790,619,1031,384);
enter(""&file_name&"",676,663);
execute("DISPLAY PUSH TOP");
selectxy(331,448,331,448);
--
execute("ANALYSIS GENERATE SELECTXY tool=cgs");
selectxy(265,374,265,374);
execute("ANALYSIS EVALUATE COMPLETE CGS_run GEN_PGM COMPILE_PGM
   RUN_PGM tooldeck=mlib/depsk_sys.tooldeck results=results
   iterations=" & xstring(iterations) & " noise_seed=" & xstring(
   noise_seed) & " ");
i := i + 1;
end while;
k := k + 1;
end while;
l := l + 1;
end while;
j := j + 1;
end while;
--
END;
END;
-------------------------------------------------------------------
--
-- Record Stopped: Wed Jul 10 09:41:21 1991
--
-------------------------------------------------------------------
```

References

[1] D. Divsalar and M. K. Simon, "Multiple-Symbol Differential Detection of MPSK," *IEEE Trans. Commun.*, vol. COM-38, pp. 300-308, Mar 1990.

[2] J. H. Winters, "Differential Detection with Intersymbol Interference and Frequency Uncertainty," *IEEE Trans. Commun.*, vol. COM-32, pp. 25-33, Jan 1984.

[3] J. C. Henry, III, "DPSK Versus FSK with Frequency Uncertainty," *IEEE Trans. Commun. Technol.*, vol. COM-18, pp. 814-817, Dec 1970.

[4] N. M. Blachman, "The Effect of Phase Error on DPSK Error Probability," *IEEE Trans. Commun.*, vol. COM-29, pp. 364-365, Mar 1981.

[5] M. K. Simon and D. Divsalar, "Doppler-Corrected Differential Detection of MPSK," *IEEE Trans. Commun.*, vol. COM-37, pp. 99-109, Feb 1989.

[6] J. H. Park, Jr., "On Binary DPSK Detection," *IEEE Trans. Commun.*, vol. COM-26, pp. 484-486, Apr 1978.

[7] J. C.-I. Chang and N. R. Sollenberger, "Burst Coherent Demodulation with Combined Symbol Timing, Frequency Offset Estimation, and Diversity Selection," *IEEE Trans. Commun.*, vol. COM-39, pp. 1157-1164, Jul 1991.

[8] W. C. Lindsey and M. K. Simon, *Telecommunication Systems Engineering*. Englewood Cliffs, NJ: Prentice-Hall, 1973.

[9] F. Couture, *Part 1: Performance Analysis of the Downlink DBPSK EHF-SATCOM System*, Contract Report, Royal Military College of Canada, Jan 1991.

[10] *SPW Block Diagram Editor User's Guide*, COMDISCO Systems Inc., Software release 2.7.

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. **ORIGINATOR** (the name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Establishment sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Defence Research Establishment Ottawa
Ottawa, Ontario
K1A 0Z4

2. **SECURITY CLASSIFICATION** (overall security classification of the document including special warning terms if applicable)

UNCLASSIFIED

3. **TITLE** (the complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S,C or U) in parentheses after the title.)

Multiple-Symbol Differential Detection of DPSK with Frequency Offset (U)

4. **AUTHORS** (Last name, first name, middle initial)

Turgeon, Martin

5. **DATE OF PUBLICATION** (month and year of publication of document)

December 1991

6a. **NO. OF PAGES** (total containing information. Include Annexes, Appendices, etc.)

46

6b. **NO. OF REFS** (total cited in document)

10

7. **DESCRIPTIVE NOTES** (the category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.

DREO Technical Note

8. **SPONSORING ACTIVITY** (the name of the department project office or laboratory sponsoring the research and development. Include the address.

DEFENCE RESEARCH ESTABLISHMENT OTTAWA
OTTAWA, ONTARIO
K1A OK2

9a. **PROJECT OR GRANT NO.** (if appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant)

Project D6470

9b. **CONTRACT NO.** (if appropriate, the applicable number under which the document was written)

10a. **ORIGINATOR'S DOCUMENT NUMBER** (the official document number by which the document is identified by the originating activity. This number must be unique to this document.)

DREO TECHNICAL NOTE 92-5

10b. **OTHER DOCUMENT NOS.** (Any other numbers which may be assigned this document either by the originator or by the sponsor)

11. **DOCUMENT AVAILABILITY** (any limitations on further dissemination of the document, other than those imposed by security classification)

(X) Unlimited distribution
( ) Distribution limited to defence departments and defence contractors; further distribution only as approved
( ) Distribution limited to defence departments and Canadian defence contractors; further distribution only as approved
( ) Distribution limited to government departments and agencies; further distribution only as approved
( ) Distribution limited to defence departments; further distribution only as approved
( ) Other (please specify):

12. **DOCUMENT ANNOUNCEMENT** (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). however, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected.)

Unlimited Announcement

UNCLASSIFIED

SECURITY CLASSIFICATION OF FORM

13. **ABSTRACT** (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

A ground terminal (GT) simulator for an extremely high frequency (EHF) satellite communications (SATCOM) system is being developed as part of the EHF SATCOM project at the Defence Research Establishment Ottawa (DREO). The GT demodulator uses *multiple-symbol differential detection of DPSK* (MSDPSK), a demodulation technique which estimates the phase reference of a DPSK modulated, burst signal over a sequence *N*. The bit error rate (BER) performance is investigated and the simulation results are presented. The BER performance of MSDPSK is shown to approach that of coherently detected DPSK (DEPSK) for large *N* and no frequency offset. The effects of frequency offset are also investigated. For a specific amount of frequency uncertainty, there is an optimum sequence size *N* that yields the best BER performance. For example, given a burst size of 13 bits and a frequency offset up to 1/50 of the bit rate, the optimum sequence size is in the range of 3 to 5 bits.

14. **KEYWORDS, DESCRIPTORS or IDENTIFIERS** (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Modulation
PSK
DPSK
DEPSK
Frequency Offset
Differential Detection

UNCLASSIFIED

SECURITY CLASSIFICATION OF FORM